Timeout Reached, Session Ends?

A Methodological Framework for Evaluating the Impact of Different Session-Identification Approaches

Dissertation

zur Erlangung des akademischen Grades

Doctor Philosophiae (Dr. phil)

Humboldt-Universität zu Berlin
Philosophische Fakultät
Institut für Bibliotheks- und Informationswissenschaft
Lehr- und Forschungsbereich Information Retrieval

von

Florian Dietz

(543203)

Der Präsident (komm.) der Humboldt-Universität zu Berlin:
Prof. Dr. Peter Frensch
Der Dekan der Philosophischen Fakultät:
Prof. Dr. Thomas Sandkühler

Gutachtende:
1. Prof. Vivien Petras, PhD
2. Prof. Dr. Robert Jäschke

Datum der Disputation: 11. August 2022

Timeout Reached, Session Ends?

A Methodological Framework for Evaluating the Impact of Different Session-Identification Approaches

Dissertation

submitted for the Degree of

Doctor of Philosophy (Dr. phil)

Humboldt-Universität zu Berlin
Faculty of Arts and Humanities
Berlin School of Library and Information Science

by

Florian Dietz

(543203)

The President (ag.) of the Humboldt-Universität zu Berlin:
Prof. Dr. Peter Frensch
The Dean of the Faculty of Arts and Humanities:
Prof. Dr. Thomas Sandkühler

Examiners:
1. Prof. Vivien Petras, PhD
2. Prof. Dr. Robert Jäschke

Date of Disputation: 11. August 2022

Zusammenfassung

Die Identifizierung von Sessions zum Verständnis des Benutzerverhaltens ist ein gängiges Forschungsgebiet des Web Usage Mining. Der Begriff wird verwendet, um die Interaktionen eines Benutzers innerhalb eines Kontexts zu beschreiben, beispielsweise die Interaktionen in einem bestimmten Zeitraum oder die Interaktionen, die auf die Erfüllung eines Bedürfnisses gerichtet sind. Unterschiedliche Definitionen und Konzepte werden seit über 20 Jahren diskutiert. Die Forschung hat gezeigt, dass die Session-Identifizierung kein willkürlicher Prozess ist und sorgfältig behandelt werden sollte. Trotz dieser Erkenntnisse ist das eher naive Session-Modell des 30-minütigen Inaktivitäts-Timeouts der Industriestandard. Diese Tendenz zu vereinfachten mechanischen Sessions anstelle komplexerer logischer Segmentierungen ist fragwürdig.

Ziel dieser Dissertation ist es zu beweisen, wie die Natur unterschiedlicher Session-Ansätze zu abweichenden Ergebnissen und Interpretationen führt. Die übergreifende Forschungsfrage lautet: Werden sich verschiedene Ansätze zur Session-Identifikation auf die Analyse und maschinelles Lernen auswirken? Dies wird durch die Untersuchung von drei Forschungsfragen beantwortet: RQ1) Wie können Sessions modelliert werden, um ein oder mehrere Informationsbedürfnisse darzustellen? RQ2) Gibt es Unterschiede in den Ergebnissen verschiedener Session-Identifikationsalgorithmen? Können diese Ergebnisse auf spezifische Identifikations- und Vergleichsmechanismen zurückgeführt werden? RQ3) Wird sich die Qualität von Algorithmen aus dem Bereich des maschinellen Lernens in Abhängigkeit von den Eingabedaten ändern? Forschungsfrage 1 wird beantworten, was benötigt wird, um ein erweitertes Session-Modell zu entwickeln, das in der Lage ist, die Informationsbedürfnisse eines Benutzers abzubilden. Es zeigt die notwendigen Methoden, um Sessions zu identifizieren und die Faktoren, die notwendig sind, um themenbezogenes Verhalten zu modellieren. Forschungsfrage 2 wird die Ergebnisse verschiedener Session-Identifikationsansätze näher erläutern und die Konzepte und ihre Ergebnisse umfassend modellieren und erklären. Forschungsfrage 3 wird die Ergebnisse der zuvor modellierten Ansätze in mehrere Anwendungen des maschinellen Lernens einspeisen und die Ergebnisse der Modelle basierend auf den unterschiedlichen Eingabedaten vergleichen.

Ein umfassender methodischer Rahmen für die Durchführung, den Vergleich und die Evaluation von Sessions wird gegeben. Die Dissertation liefert Implementierungsleitlinien für 135 Session-Identifikationsansätze am Beispiel eines kompletten Jahres (2018) von Daten der Benutzer einer deutschen Preisvergleichs-E-Commerce-Plattform. Der Datensatz umfasst 1.268.619.378 Interaktionen von 78.361.923 Benutzern. Die Umsetzung umfasst mechanische Konzepte, eine Vielzahl logischer Konstrukte und die Kombination mehrerer Mechaniken. Es zeigt, wie logische Sessions aus Benutzersequenzen konstruiert werden, indem Embedding-Algorithmen (word2vec) in Interaktionsprotokollen verwendet werden. Hierbei wird ein neuartiger Ansatz zur Identifizierung logischer Sessions verfolgt, indem die thematische Nähe von Interaktionen anstelle von Suchanfragen allein verwendet wurde. Anhand von 17 ausgewählten Kennzahlen werden alle Ansätze verglichen und quantitativ beschrieben, wobei Vor- und Nachteile sowie Besonderheiten hervorgehoben werden. Eine Auswahl von 26 Ansätzen wird als Eingabedaten für drei Aufgaben des maschinellen Lernens verwendet (Word-Embeddings, Item-Recommendation, exploratives Benutzer-Clustering). Diese Anwendungen sollen zeigen, dass die Verwendung unterschiedlicher Sessions als Eingabedaten einen deutlichen Einfluss auf das Ergebnis hat.

Der Hauptbeitrag dieser Dissertation besteht darin, einen umfassenden Vergleich von Session-Identifikationsalgorithmen bereitzustellen. Diese Forschungsarbeit bietet eine Methodik zum Implementieren, Analysieren und Vergleichen einer Vielzahl von Mechanismen, die es ermöglichen, das Benutzerverhalten aus vielfältigen Perspektiven zu verstehen, und es Systembesitzern und Forschern ermöglicht, die Auswirkungen ihrer Session-Modellierung besser nachzuvollziehen. Eine Methode wird vorgestellt, um logisch verbundene Sessions zu identifizieren, die die Informationsbedürfnisse von Benutzern auf der Grundlage von Sequenzeinbettungen darstellen. Hierbei wird thematische Nähe verwendet, um die Interaktionsähnlichkeit abzuschätzen. Das Konzept erlaubt unterschiedliche Ebenen der Komplexität, indem es verschiedene Vergleichsmechanismen und -kontexte einführt. Die Hauptergebnisse zeigen, dass unterschiedlich strukturierte Eingabedaten die Ergebnisse von Algorithmen oder Analysen drastisch verändern können, was beweist, dass die Session-Identifikation mit Sorgfalt erfolgen und als integraler Bestandteil des Preprocessings etabliert werden sollte. Es ist anzuraten, mehrere Session-Modelle zu verwenden, um das Benutzerverhalten zu verstehen: Die Unterschiede in den Ergebnissen verschiedener Session-Konzepte sind bemerkenswert und sollten nicht ignoriert werden.

Abstract

The identification of sessions as a means of understanding user behaviour over time is a common research area of web usage mining. The term is used to describe the interactions of a user within a certain scope, for example, the interactions in a certain time period or the interactions directed towards the completion of a specific need. Different definitions and concepts have been discussed for over 20 years. Research has shown that session identification is not an arbitrary task and should be handled with care. Despite such findings, the rather naive session model in the form of a 30-minute inactivity timeout is the industry standard. This tendency towards simplistic mechanical sessions instead of more complex logical segmentations is questionable.

This dissertation aims to prove how the nature of differing session-identification approaches leads to diverging results and interpretations. The overarching research question asks: will different session-identification approaches impact analysis and machine learning tasks? This is answered by investigating three research questions: RQ1) How can sessions be modelled to represent one or more information needs? RQ2) Are there differences in the results of different session-identification algorithms? Can these results be attributed to specific identification and comparison mechanics? RQ3) Will the performance of machine learning algorithms change depending on the input data? Research question 1 will answer what it takes to develop an enhanced session model that is able to represent the information needs of a user. It will show the necessary methods to identify sessions and the factors that are involved to model topically-related behaviour. Research question 2 will elaborate on the results of different session-identification approaches, modelling and explaining the concepts and their outcomes. Research question 3 will feed the outcomes of the previously modelled approaches into multiple machine learning applications, comparing the outputs of the models based on the differing input data.

A comprehensive methodological framework for implementing, comparing and evaluating sessions is given. The dissertation provides implementation guidelines for 135 session-identification approaches utilizing the example of a complete year (2018) of user traffic data from a German price-comparison e-commerce platform. The dataset includes 1,268,619,378 interactions from 78,361,923 users. The implementation includes mechanical concepts, a variety of logical constructs and the combination of multiple mechanics. It shows how logical sessions were constructed from user sequences by employing embedding algorithms (word2vec) on interaction logs; taking a novel approach to logical session identification by utilizing topical proximity of interactions instead of search queries alone. Using 17 selected measures, all approaches are compared and quantitatively described, highlighting advantages, disadvantages, and peculiarities. A subset of 26 approaches is used as input data for three machine-learning tasks (word embeddings, item recommendation, exploratory user clustering). The application in these tasks is intended to show that using different sessions as input data has a marked impact on the outcome.

The main contribution of this dissertation is to provide a comprehensive comparison of session-identification algorithms. This research provides a methodology to implement, analyse and compare a wide variety of mechanics, making it possible to understand user behaviour from manifold perspectives and allowing system owners and researchers to better understand the effects of their session modelling. It introduces a method to identify logically connected sessions that represent the information needs of users based on sequence embeddings, utilizing topical proximity to estimate interaction similarity. The concept allows different levels of complexity by introducing various comparison mechanics and contexts. The main results show that differently structured input data may drastically change the results of algorithms or analysis, thereby proving the point that session identification should be done with care and established as an integral part of the preprocessing. It may be well advised to use multiple session models to understand user behaviour: the divergences in the results of different session concepts are quite notable and should not to be ignored.

Danksagungen

Ich möchte mich bei allen Menschen bedanken, die mich in den letzten Jahren durch dieses Kapitel begleitet haben. Vielen Dank für die Unterstützung auf diesem Weg.

Besonderer Dank gilt an dieser Stelle meiner Doktormutter Vivien Petras. Vielen Dank für Deine kompromisslose Förderung und Dein unerschütterliches Vertrauen, ganz egal, welche neuen Widrigkeiten ich berichten musste. Die dichte, vertrauensvolle und immer motivierende Betreuung hat nicht nur zur Qualität der Arbeit, sondern auch enorm zu meiner persönlichen Entwicklung beigetragen. Hervorheben möchte ich die konstruktiven Diskussionen und Dein Auge für Details in all unseren Sitzungen, die mich stets in die richtige Richtung gelenkt haben. Du hast meine Leidenschaft für Evaluation und Information Retrieval, Nutzerverhalten und kritisches Hinterfragen entfacht und so nicht nur meine akademische, sondern auch meine berufliche Laufbahn entscheidend beeinflusst. Vielen, vielen Dank! Bedanken möchte ich mich auch bei Robert Jäschke. Die Bereitschaft, in ein derart ausformuliertes Thema einzusteigen und die Hinweise und Ratschläge auf die kleinen, fiesen Detailfragen haben einen entscheidenden Teil der Arbeit deutlich erleichtert.

Meinen Kollegen und meinem Team bei idealo möchte ich ebenfalls danken. Besonders hervorheben möchte ich Holger als die Person, die meine Vorliebe für Daten erkannt und sorgsam geschult hat und durch das Vertrauen in mich und die Offenheit für Neues diese Arbeit in dieser Firma erst möglich gemacht hat. Ebenso bedanken möchte ich mich bei Katharina, die mir insbesondere das letzte Jahr über stets das Vertrauen entgegengebracht und den nötigen Freiraum ermöglicht hat, um die Arbeit letztlich abzuschließen. Danke auch David, für die allgegenwärtige Bereitschaft, irgendwelche Themen zu diskutieren und meine Gedanken kritisch zu hinterfragen, und Max, für die Freitagnachmittage, an denen wir lauter neue Ideen diskutiert haben.

Außerdem möchte ich mich bei meinen engsten Freunden bedanken. Vielen Dank an Thomas, für die gediegene Alltagsgestaltung sowie die nötige Ablenkung. Theresa für das immer offene Ohr und Deinen unerschütterlichen Glauben an meine Fähigkeiten. Bartosz, der nicht nur zum Mentor, sondern auch zu einem meiner besten Freunde wurde.

Danke auch an meinen wunderbaren Kater Joker, der mit seiner Ruhe und Gelassenheit meine Ausgeglichenheit gesichert hat und an meine Familie für den Rückhalt. Zuletzt möchte ich mich auch aus ganzem Herzen bei Franzisca bedanken. Die emotionale Unterstützung und das immer offene Ohr, wenn ich über ein neues Problem geflucht habe, werde ich nie vergessen.

Contents

List of Abbreviations and Acronyms

AM	Arithmetic Mean
API	Application Programming Interface
AWS	Amazon Web Services
B-R	Bounce Rate
BM25	(Okapi) Best Matching; general retrieval algorithm
BN	Bayesian Network
CBOW	Continuous bag of words
CRM	Customer Relationship Management
CSTM	Complex Search Task Model
CTAS	Create Table as select
CV-R	Conversion Rate
DBSCAN	Density Based Spatial Clustering of Applications with Noise
EAN	European Article Number
EMR	Elastic MapReduce
ERR	Error Rate
ESA	Explicit Semantic Analysis
ETL	Extract, Transform, Load
GDPR	General Data Protection Regulation
HDBSCAN	Hierarchical Density-Based Spatial Clustering of Applications with Noise
HR	Hit Rate
HTTP	Hypertext Transfer Protocol
IIR	Interactive Information Retrieval
IP	Internet Protocol
IR	Information Retrieval
JSON	JavaScript Object Notation
KPI	Key Performance Indicator
LDA	Latent Dirichlet Allocation
LO	Lead-out
LOD	linked open data
LSTM	Long Short-Term Memory

MRR	Mean Reciprocal Rank
NLTK	Natural Language Toolkit
ODP	Open Directory Project
OS	Operating System
PCA	Principal Component Analysis
Pi	Pageimpression
R	Recall
RNN	Recurrent Neural Network
SD	Standard Deviation
SEM	Search Engine Marketing
SEO	Search Engine Optimization
SER	Slot Error Rate
SERP	Search Engine Result Page
SQL	Structured Query Language
SVM	Support Vector Machine
TCP	Transmission Control Protocol
URL	Uniform Resource Locator
UTM	Urchin Tracking Module
WMD	Word Mover's Distance
ac	Technical identifier used for the 'all complete' comparison context
ad	Technical identifier used for the 'all direct' comparison context
bm25	Technical identifier used by the author in his experiments; meaning the 'bm25' logical session approach
bm25ti	Technical identifier used for the 'bm25 with temporal inactivity' combined session approach
cc	Technical identifier used for the 'consecutive complete' comparison context
cd	Technical identifier used for the 'consecutive direct' comparison context
ean	Technical identifier used for the European Article Number
geom	Technical identifier used for the 'geometric' combined session approach
iRNN	Technical identifier used for the 'Inter-Session Recurrent Neural Network' by Ruocco et al.[225]
id	Identifier
knn	Identifer for the k-nearest Neighbours algorithm
l	Technical identifier used for the 'lexical' logical session approach
lti	Technical identifier used for the 'lexical' with temporal inactivity' combined session approach
noi	Number of Interactions
pRNN	Technical identifier used for a (Plain) Recurrent Neural Network
sid	Session Identifier
td	Technical identifier used for the 'temporal dynamic' (inactivity) mechanical session approach
tf	Technical identifier used for the 'temporal fixed' mechanical session approach

tf-idf	term frequency-inverse document frequency
ti	Technical identifier used for the ‘temporal inactivity’ mechanical session approach
u2v	Technical identifier used for the ‘user to vector’ logical session approach
u2vti	Technical identifier used for the ‘user to vector with temporal inactivity’ combined session approach

List of Figures

List of Tables

Chapter 1

Introduction

E-commerce is all about understanding user needs. System owners strive to understand what their users are doing, how often they visit the system and to what extent individual users or user groups contribute to the system's success. Measuring user activity is a key component. Only by having information about the traffic on the system is it possible to understand and even measure its performance.

User activity in the form of logged interactions, that is, browsing or querying a system, is segmented into an artificial construct called a session. The term, assumed to be first coined in 1995 by Catledge and Pitkow [43], is used to describe the interactions of a user within a certain scope, for example, the interactions in a certain time period or the interactions directed towards the completion of a specific need. The aims of session detection or session identification is to find the most suitable boundary for dividing a user's interactions into such grouped segments.

Where interactions like queries or a click on a page are the fundamental units of user behaviour, sessions are the glue that connects these units together. This picture can be drawn even more figuratively: interactions are the footprints of users while sessions can be seen as the path a user takes through a system. A system owner can use sessions to understand where the user is heading to or what path s/he[1] is taking on a system. The question is what a session actually is. Their nature is abstract: there is no clear general definition, the boundaries of sessions are unclear and there is no common standard capable of representing user behaviour.

Instead, there are many different variants and terminologies of sessions. The most common session-identification approach is using an inactivity timeout, with the well-known and irritating 30 minutes being the common industry standard. Here, a session connects all of the user's interactions within 30 minutes of each other: after a break of 30 minutes or more, a new session will start, in the assumption that the inactivity break implies that the user has started work on a new topic. As described in [62], this 30-minute inactivity rule is being used very generously in many applications and systems without much consideration. Along the inactivity timeout, other variants may use a maximum-time-per-session or

[1] The pronoun 'they' is used interchangeably throughout along with 's/he' to indicate users regardless of gender or the gender the user identifies with.

try to reproduce the user path on the system by connecting interactions via Uniform Resource Locators (URL) and Hypertext Transfer Protocol (HTTP) referers. Other session types work differently: instead of using a mechanical rule such as the time between interactions, the so-called logical sessions or task-based sessions employ completely different identification-comparison methods, stepping away from purely mechanical boundaries.

The idea behind these logical sessions as described by Jones and Klinkner [120], Gayo-Avello [80] and Hagen et al. [88, 89] is that time is just not enough to differentiate between user activities. Instead, topical similarity is analysed to connect interactions, not only between consecutive interactions but also by assuming that a user may work on a topic in a start-stop-start manner, stopping to work on a different topic only to start again after some time. This enables completely different session constructs. Users may search for one thing, browsing for something else in another tab, and then continue their previous search – these sessions account for multitasking behaviour and view each interaction in the light of previous interactions [120, 131, 133].

In the majority of examples in the literature, topical similarity is completely dependent on the existence of queries. The authors try to identify search sessions by connecting related queries and their associated clicks to any search results. Many different proposals on how to identify query similarity have been made, either utilizing lexical or semantic distance measures. Examples from the literature use distance metrics (Levenshtein, Jaccard or Cosine) between words [88, 89, 120, 144, 215, 231], characters or n-grams [80, 120, 209, 254], some even employ external knowledge databases to expand queries with semantically related terms [88, 89, 106] while others compare the overlap in search results of queries [88, 89]. Newer works employ more sophisticated methods on top of these, such as clustering on the calculated pair-wise distance [154, 297, 298] or calculating distances between the vector representations of query strings [181, 229, 262]. Relying solely on queries may be a problem though. Not every information system relies on users querying its contents. Navigational elements and simple browsing could be the main ways of accessing a system. Having no or few queries in the data, logical sessions cannot be constructed as suggested. A different session modelling is needed.

Many of these examples still use a temporal mechanical boundary to identify the sessions they are then using to extract tasks and logical sessions. Still, they present valuable input to verify that there is indeed a difference between the mechanical segmentation of consecutive interactions and the topical connection of all related interactions. Nevertheless, the mechanical sessions seem to be omnipresent despite the many examples. As Hagen et al. describe, this is a problem: for many use cases, it would be much more reasonable to apply logical boundaries to user activity than mechanical rules [89].

This is logical. A mechanical rule such as the inactivity timeout may group user behaviour differently than connecting, for example, queries based on the same topic. One is a strict rule that simply groups interactions under the assumption that a timeout will work globally for all users, whereas the other actually follows through with the assumption that related interactions should belong to the same session. This divergence is likely to lead to different results. Still, the majority of research simply uses inactivity timeouts without even considering the implications of that decision [62].

Sessions are used for everything. They are usually the most atomic unit by which user behaviour is looked at. Be it basic web analytics, recommendation scenarios or search improvement tasks, the aggregation level for the input data is usually a session. Having the 30-minute industry standard here is kind of a mixed blessing. On the one hand, having every system use the same session boundary ensures comparability between these systems. But this also implicates that all systems and all users of these systems behave in the same way. This is intuitively wrong: someone browsing a news articles page will behave rather differently from someone checking emails or planning a vacation. Many examples prove these variances in behaviour by using different parameters for different applications [225].

This is also true within the same system. Not all users will behave in the same way. Even though using the same rule for everyone would again ensure comparability across the system traffic, doing so may pollute information about individual user behaviours. For example, it can easily be assumed that users shopping for a new television will browse or search differently compared to users trying to find replacement parts for their washing machine. With a global mechanical rule and therefore a globally defined mechanical session, this difference would probably not be reflected, whereas a logical mechanic may be able to represent it.

1.1 Motivation: The One To Rule Them All?

The differences between the mechanics of different session-identification methods are assumed to be quite drastic. It has been shown in multiple publications that even different timeouts influence the depicted behaviours or are not that reliable [51, 83, 184, 192]. Other publications show that different timeouts do not change session-level metrics significantly [33, 146]. Any differences might be amplified by completely different session mechanics. Many applications are assumed to benefit from another session mechanic.

Nevertheless, apparently the majority of researchers still use the 30-minute inactivity rule or some other form of inactivity timeout [23, 62]. Simply using the same inactivity rule time and again without due consideration is too simplistic. There is no general consensus that the 30-minute inactivity rule or even any temporal constraint might deliver good results. It is a simplistic estimation that is used without acknowledging the possible side effects. The reliability of web usage mining, and the quality of any findings achieved through it, is heavily dependent on proper preprocessing and preparation of the data used as input. Many works consider data preprocessing as either essential or the very first step in any project involving interaction logs [10, 51, 66, 75, 98, 238]. This is especially true for the identification of sessions. Careless session modelling may lead to invalid patterns, faulty preprocessing or ambiguous results and therefore wrong conclusions, depending on the system and application. The results of any analysis or further processing, as in data mining or machine learning applications, is dependent on how the data is preprocessed. If it is incomplete, erroneous or noisy, there may be false interpretations. This dissertation is motivated by this apparent lack of forethought. Session identification should be regarded as an integral part of the preprocessing of any dataset before applying it as input data to any application.

Different sessions are assumed to lead to varying interpretations of user behaviour. Depending on what method is used to identify sessions, different segments of interactions will form, potentially resulting in completely different assumptions regarding that behaviour [51, 83, 188]. This research will show that it is indeed not unproblematic to simply use an inactivity timeout, proving the points made in [62]. The session-identification method has a strong effect on the outcome of not only the sessions identified in the dataset, but also on the results of algorithms utilizing these sessions as input data.

The motivation comes from the observation that while the 30-minute inactivity timeout is the most prevalent session approach in use today, apparently logical mechanics may offer a marked improvement compared to that rule. Using the same rule for all algorithms and systems dangerously simplifies the problem. The 30-minute inactivity timeout might be perfectly fine for most applications – but arbitrarily and unquestioningly applying it as the basis for any and all algorithms may lead to incorrect conclusions, no matter the quality of the algorithm[2]. Just because it has been standard for more than 25 years, it should not be assumed to be correct.

This dissertation aims to prove how the nature of differing session-identification approaches will lead to different results and interpretations. By providing a methodological guide to compare different session models and offering a comprehensive explanation for the implementation of variants of each described session-identification method, this dissertation gives an estimation of the impact of these different approaches, thereby answering the big question of whether or not the session-identification approach makes a difference in log file analysis and the application of log files in machine learning tasks.

1.2 Research Questions

There is an assumption that user behaviour as represented in sessions looks very different depending on how it is modelled. This dissertation will investigate whether the modelling and analysis of different session approaches changes the understanding of user behaviour and assumed user information needs when interacting with an e-commerce platform. It will research how different session-identification algorithms change the structure of identified sessions and therefore change the output from any downstream algorithm using these sessions as input data. The required steps and methodology for establishing a concept of session models and comparing their results are demonstrated, including traditional timeout sessions along with sessions built with the intention of representing an information need. An objective evaluation of these algorithms is presented. As outlined in Section 1.1, the overarching research question aims to answer whether different session-identification approaches impact log file analysis and machine learning tasks. This can be drilled down to multiple research questions:

[2] Parts of this were already published in [62].

RQ1 How can sessions be modelled to represent one or more information needs?

RQ2 Are there differences in the results of different session-identification algorithms? Can these results be attributed to specific identification and comparison mechanics?

RQ3 Will the performance of machine learning algorithms change depending on the input data?

Research question 1 will answer what it takes to develop an enhanced session model that is able to represent the information needs of a user. It will show the methods and algorithms that are necessary in order to stitch sessions together and the factors that are involved to model topically-related user behaviour. Research question 2 will elaborate on the results of different session-identification approaches, comprehensively modelling and explaining the concepts and their outcomes. Research question 3 will feed the outcomes of the previously modelled approaches into multiple machine learning applications, comparing the outputs of the models based on the differing input data.

1.3 Case Study

A case study is conducted in order to answer the research questions and prove the feasibility of the methods. Using the log data of a German price comparison platform with several million logged events per day, the research questions are approached and tested with the data of a real-life production system. The website is structured in a similar way to a classic shopping website. The homepage is the root of a large tree with multiple broader categories like electronics or fashion. These root categories become more specific as they branch out to present the product categories, containing various product pages with information about the products the shop offers and which are used for comparing prices. Inside a product category, product pages are listed according to popularity. On a product page, users can find detailed product information, product images and a list of prices. They can perform various actions like putting the product on the wish list, setting a price alert for the product or clicking through to a particular shop's page. As the website has a full range of products, user affinities related to categories could be another relevant factor for the data model.

1.4 Research Design

This dissertation provides a comprehensive comparison of 135 session-identification approaches. It is designed as a full-scope evaluation framework, explaining preprocessing from preparing the raw dataset, to modelling various session approaches, as well as an experimental evaluation in the form of an exhaustive analysis of all session approaches, and their application in three different machine learning use cases.

The first step is to actually create a usable dataset. Using a simplified user concept, a dataset of user interactions is extracted from the log files of the price comparison platform and enriched with further needed identifiers. Additionally, the dataset is filtered to remove any users that only have one interaction with the system. Further preprocessing steps such as normalizing string values and creating additional features by utilizing the existing fields were also done. Finally, the result is a raw dataset with 1,268,619,378 interactions from 78,361,923 users.

The resulting dataset is then used to model a variety of 135 session approaches. In a comprehensive methodological overview, detailed steps for implementing sessions of eight different archetypes are shown. A complete step-by-step process is developed in how to model logical sessions without having to rely on search queries as most examples from the literature do.

Additionally, two ways of evaluating the resulting sessions are described. The evaluation is intended to prove a) differences in the resulting sessions and b) differences in the output of applications fed with these sessions as input data. Multiple measures are given that are normally used to evaluate system performance. The design of the evaluation is intended to highlight differences in the results identified by the different session-identification approaches. The actual quality and correctness of the identified sessions is acknowledged in parts but not as the main focus, as it is not the main topic of this dissertation.

The first part of the evaluation is a thorough analysis of the complete sessionized dataset, comparing all session approaches across 17 selected measures. These measures, such as the number of sessions, the number of interactions per session or the number of sessions with only one interaction (i.e. the bounce rate), could be used to evaluate system performance. The goal here is to show that varying session-identification methods will lead to different observations, potentially leading to misinterpretations of a system's performance.

The second part of the evaluation uses a sample and a selection of 26 session approaches as input data for three machine learning applications. This part of the evaluation highlights the observations made in part one. The application in the three machine learning use cases – analysing category tree similarity based on word embeddings, a recommendation scenario as well as an exploratory clustering – is intended to show that using different sessions as input data has a marked impact on the outcome of machine learning tasks. By comparing the results per session approach, the differences are once again highlighted. Additionally, the results seen here can be viewed as an initial glance at the potential qualitative differences between the identified sessions.

1.5 Organization of the Dissertation

The dissertation is organized as follows. Chapter 2 provides a comprehensive overview of the current state of session identification. The literature review introduces all relevant concepts: mechanical sessions using time or structural mechanics as well as logical sessions using lexical or semantic similarity. A quick detour around why the mechanical inactivity timeout is omnipresent is given. Additionally, the dissertation provides an overview of the common evaluation procedures in session identification as well as presenting in brief the manifold applications using sessions as input data.

Chapter 3 introduces the price comparison platform providing the log files used in this dissertation. The system structure is explained, giving information about the website and how the business model is implemented. Additionally, a simplified overview of the tracking architecture is given and potential issues are discussed.

In Chapter 4, the complete methodological framework is outlined. First, a concise terminology for sessions is proposed, unifying the various assumptions made in the literature. The dataset is modelled from the log files using the newly introduced concepts and different preprocessing steps. Afterwards, the necessary steps to implement the session-identification approaches tested in this research are given in detail. The last part explains the proposed evaluation framework.

The first part of the evaluation is presented in Chapter 5. Here, a descriptive analysis explains the specifics of the dataset. After that, all 135 session approaches are thoroughly analysed. The differences are highlighted and some interpretative observations are given. The results are then summarized and discussed.

Chapter 6 employs the identified sessions as input data in three different machine-learning use cases, again highlighting the observed differences in the outcome and discussing the results. Implemented are sequence embeddings, a recommendation scenario and a clustering task.

Finally, Chapter 7 summarizes the most important findings again along with the main contributions provided by this dissertation. Additionally, potential limitations are discussed. Some suggestions for future work are presented. Finally, the dissertation is concluded with a few closing words.

Chapter 2

Modelling Interactions with Information Systems from Interaction Logs

This chapter outlines the theoretical foundation for the dissertation. First, a short overview of working with interaction logs in general and modelling sessions specifically is given. The reasons for the identification of sessions are presented in Section 2.1. Afterwards, relevant definitions and their limitations are introduced in Section 2.2. Using the definitions, Section 2.3 concentrates on the various approaches used to identify sessions in interaction logs. A distinction is made between mechanical and logical sessions. Methods are distinguished by type, according to the nature of the factors used, and not necessarily by algorithm. Various approaches along with the respective conceptual models are presented. In Section 2.4, the commonly applied evaluation methods are explained. Section 2.5 presents a short overview of the application of sessions in actual use cases.

2.1 Using Sessions to Understand User Behaviour

Web usage mining has been around for quite some time now. Since the late 1990s and early 2000s, researchers have contributed dozens of studies about handling interaction logs and how to utilize them in their respective fields of research. These early studies focus on user behaviour and the interpretation of how users interact with search systems [18, 43, 111, 112, 240, 244, 246, 280]. Initial findings provide insights into the average number of queries and reformulations, the average number of query terms, the number of result-page visits and type of query [31]. Web usage mining allows researchers to better understand user behaviour and the relationship between the user and the system. A precondition is the segmentation of interaction logs into measurable units. Often referred to as the identification of sessions, the aim of this process is to find meaningful representations of a user's history. The term session is used arbitrarily; when speaking of the identification of sessions, the extraction of segments defined in any way is meant. There are many different approaches to sessionizing logs, which are more closely defined in Section 2.2.

However, initially, it is important to understand the reasons behind the need to group the interactions of users into different segments; for example, to understand user needs and user behaviour, evaluate system performance or simply support users and systems.

The evaluation of information systems is no longer a simple model of topics, system and document collection. In fact, user behaviour has to be analysed as a whole in order to understand the user's interaction with the system. From formulating a set of queries for a specific topic to judging the relevance of documents to query refinement: the whole process needs to be captured and transformed into metrics for evaluating how the system deals with problematic encounters experienced by the user or vice versa. A system or any application needs a reasonable measurable unit to represent user behaviour. Only then will the system be able to support users who get stuck, learn query reformulation patterns or understand user behaviour at all [89]. The last point is particularly important. Were a system to have no representation of user behaviour in the form of sessions, it simply would not be able to keep a measurable track of the user's activities, leading to a concentration on single interactions. Originally, this was more or less the case in traditional Information Retrieval (IR). The Cranfield Paradigm compares the retrieval performance of different systems on a set of documents with a set of topics (queries) [169]. Cranfield-style evaluations are useful in the development and initial tuning of retrieval algorithms, but the data preparation and collection is expensive, the coverage is limited and there is an inherent bias detected among participating users, judges and collections [78, 169].

The limitations and concentration on the system's retrieval performance is problematic since it loses sight of the user. For example, users may issue the same query and mean different things with it. A system confronted only with the individual topic would not be able to detect any difference and rank the documents the same way, as there are no contextual features used for measuring relevance. Furthermore, (real) users might need multiple queries to fulfil an information need. When evaluating only individual queries to retrieve documents, this might lead to a poor representation of the user's satisfaction in terms of retrieved relevant documents. The same is true for users at different stages of fulfilling their information needs; when issuing the same query in the early and late stages during the process of finding information, a user may expect the results to reflect the difference in the knowledge they have gained [147]. With the prevalence of the Internet in all areas of life and daily information seeking, these problems become even more immanent [270]. Modelling sessions to overcome these limitations to gaining better understanding of user behaviour over an extended period of time and throughout the whole interaction with a system, is the logical evolution from the traditional Cranfield Paradigm for evaluation.

When considering the specifics of user behaviour, session identification becomes even more important. In order to deal with specific behavioural patterns, systems need to be able to recognize them. This is not possible when focusing on single interactions. A representation of interactions within their respective context is needed to be able to identify and, possibly, react to a user's specific behaviour. Behavioural patterns like multitasking and interleaving interactions as well as more complex search tasks require specific session modelling.

Searching for information and interacting with an information system is often a multitasking process [243]. The fact that multiple tasks are worked on in the same visit to the information system makes it particularly hard to find proper session boundaries. Spink et al. [245] argue that search technologies are usually designed to accommodate users searching sequentially. The authors analysed sessions with two queries and sessions with three or more queries to find out to what degree these queries contain multiple topics. They found that 81% of the two-query sessions and 91.3% of the three-or-more-query sessions show multitasking patterns. The sessions contain many different topics, frequently changing depending on the number of queries.

Mehrotra et al. [177] report that multitasking behaviour is fairly common and should be acknowledged in personalization systems. The authors emphasize that there are strong differences in behaviour depending on the user disposition towards multitasking, topic or even interest. According to Donato et al. [67], 10% of search sessions contain queries belonging to longer-lasting information needs, making up 25% of the overall query volume. This requires a user-centric segmentation of the interaction logs. Other studies second these results [7, 70, 141, 142, 147, 163, 164, 165, 181]. The need for user-centric sessions is even stronger when considering interleaving tasks [120, 133, 243, 245]. Interleaving tasks, where the user works on a task, stops and works on another task, then starts again with the initial one, require a sophisticated representation in order to support the user.

Another specific behavioural pattern relates to longer-lasting information needs. Users working on fulfilling a need may come back multiple times [266]. This type of behaviour is reflected in the field of 'complex search' [13, 19, 218, 275]. 'Complex search' assumes that the user has different subtasks when working on an overarching task, because a complex search task is too hard to accomplish with a single query [226]. A frequently used example of a complex search task is the planning of a vacation. The user might have to look for flights, book a hotel and look for activities in the target location, which all represent potentially dissociated subtasks. It gets even more complicated when considering that different stages of information-need fulfilment might also have an impact on how a user behaves [13, 19, 46, 214, 222, 227, 265, 275, 279, 301].

Somewhat similar are regularly recurring information needs and longer-lasting dependencies between the interactions of a user. Not only recent queries but also long-term behaviour and interests are important for understanding information needs [67, 159, 222, 225]. This could be a user who bought a laptop fairly recently. S/he will probably not be interested in recommendations regarding another new laptop, but may be receptive to information about accessories. Another example is a user who visits a system regularly for the same information, be it for the weather forecast, news or football results.

Without any segmentation of the user's interactions, all of these different behavioural patterns will hardly be unidentifiable. A system that does not understand the behaviours and interests of users cannot cater for their needs. This clearly indicates the need for an adapted segmentation that enables the system to better support and understand the user. Evaluating system performance would be more realistic. Any measurement of performance will be more closely adjusted to the estimation of user satisfaction and therefore better than not considering the user at all. Different segmentation concepts enable system owners to gain measurements of how the system performs; both in terms of behaviour and satisfying a user's need.

If the system is able to reflect the user by segmenting the logs in a reasonable way, the information hidden in this representation can be used to a) support the user and b) estimate how satisfied the user is. This is done by utilizing implicit feedback extracted from the segmented interaction logs instead of explicit feedback collected directly from the user. By modelling sessions in a way that caters to the application or the system in general, implicit feedback can greatly improve system performance and user experience by supporting the user in their identified task. Systems may be able to better understand the needs and problems of a user by utilizing features such as clicks or dwell time. This assumption is supported by multiple studies [3, 78, 118, 129, 149, 301]. Implicit features differ from user to user and from task to task [127]. This is reasonable considering that studies have shown different types of shopping behaviour [1, 214, 224, 241, 275]. Interaction patterns are not uniform and therefore any boundary used for sessions has to be understood and considered [289]. Every interaction might have a different context and differing implications. Differing search behaviour patterns are another indicator of why meaningful sessions are needed.

The advantages of segmenting user interaction logs into sessions are manifold. Only with the right session-detection strategies is it possible to identify and use all of the above-mentioned elements. Therefore, the modelling of a user representation is of fundamental importance. Why sessions are needed is obvious. The type of sessions and how these are actually modelled is less clear. The following section will take a closer look at the different units of measurement.

2.2 Defining Sessions

The identification of sessions is an ambiguous problem. There are multiple possible definitions of how to segment a user's history, depending on the targeted application [116]. Because there are many different systems, applications and theories, there are various definitions of sessions. The lack of standards and the resulting need for interpretation is an imminent problem identified by this research and other research [120]. The need for interpretation leads to a bouquet of overlapping definitions. The result is a research environment with no clear terminology, incomparable results and no general consensus.

Initially, the term session was used as a surrogate for all events of users interacting with an information system during a single visit [43]. The lack of reliable user identifiers might be the reason for this type of segmentation, whereby events in close temporal proximity are assumed to be connected and are thus grouped together as if they were from the same user. This understanding of sessions as an individual visit from an individual user was prevalent in the early days; all consecutive queries (or interactions) without a significant time gap were considered a (mechanical) search session [88].

Fundamentally, there are two loose concepts for session identification: 1) mechanical sessions, which are segmented by a purely mechanical boundary and 2) logical sessions, also referred to as tasks, missions or episodes, which aim for a segmentation based on the underlying information need or task. Mechanical sessions are easily detectable but limited to basic statistical analysis [88], whereas logical sessions are assumed to allow a much more detailed and sophisticated representation of a user.

The early methods all aimed for the detection of mechanically divided sessions even without explicitly saying so. Hence, most of the early works rely on purely time-based constraints: a) two consecutive queries belong to the same segment whenever the time elapsed is smaller than a defined threshold or b) a query belongs to a session if the time elapsed between it and the initial starting event is within a defined time period. Many different time gaps have been experimented with: five minutes [68], 10–15 minutes [95], 30 minutes [68, 215], 60 minutes [33] or even 120 minutes [33].

From this point of view, the term session describes just a mechanically divided interaction log. There is no real assumption of user behaviour and no real definition yet, the term session solely refers to the collected requests by a user during one individual visit [238]. The first actual definition of the term comes from Silverstein et al. [232], where they define 'a session as a series of queries by a single user made within a small range of time. A session is meant to capture a single user's attempt to fill a single information need' [232, p. 7]. Here, sessions are supposed to capture a single information need, which somewhat contrasts with a mechanical boundary. Similarly, Jansen et al. [113] define a session as 'a series of interactions by the user toward addressing a single information need' [113, p. 862]. Other studies take up this definition again by defining a session as a sequence of queries within a specific time frame [28, 123].

He and Göker [95] assume that a growing temporal gap is an indicator of topic change. They want sessions to group all events related a) to an evolving information need and b) through proximity in time. Therefore, session start and end would indicate topical change. This is further built upon in He et al. [96] by bringing in the concept of session shift and session continuation, where two successive queries might indicate a new session depending on how they relate to each other. Technically, this is already a shift from mechanically divided segments to a logical construct, although with some weaknesses. Temporal proximity is the first boundary, followed by a logical boundary in deciding if a new query denotes a shift or a continuation. A similar construct is proposed by Yu et al. [290] and Radlinski and Joachims [215], who connect queries according to their reformulations.

Stricter concepts were tested as well [52, 128, 153]. Here, a (query) session consists only of a query and its subsequent behaviour. This is a limited construct. There is no logical connection between consecutive queries. Also, there is no possibility to associate any subsequent behaviour with a certain query.

Other concepts presented over time take up on the concepts of multitasking and complex search. These naturally evolved from the concept of a mechanical session and represent the notion of a logical session. The sessions they describe continue the idea of having a segmentation aiming for one or more information needs. A recent representative definition comes from Gomes et al. [83], who state that ‘a session is a sequence of activities followed by one individual to satisfy an information need, regardless of the elapsed time, number of interactions with the system, or the existence of interruptions on these interactions’ [83, p. 185]. The session is defined as the atomical information unit centred on an information need regardless of other factors. In addition, it accounts for interleaving and multitasking by connecting all events related to one information need through time and space.

The term session is obviously very misleading, as it mixes up different levels of various concepts. Frequently, the research does not state which definition is being followed. What is worse is that mechanically detected sessions are often utilized even though a logical session would be much more reasonable [88]. This has resulted in new terms and definitions being introduced to the research. Jones and Klinkner [120] introduce search goals, which are defined as the atomic information need resulting in one or more queries. This definition is identical to other session definitions. The authors also present search missions, connecting multiple related information needs in an overarching concept combining multiple search goals. A search session in their view is all user activity within a certain time window. They argue with historical ambiguity: a session was ‘simultaneously (1) a set of queries to satisfy a single information need (2) a series of successive queries, and (3) a short period of contiguous time spent querying and examining results’ [120, p. 700].

More recently, the term task was coined for this type of concept. Maguitman [166] defines tasks as ‘a piece of work required to achieve an objective’ [166, p. 2]. This loose definition resembles that of Gomes et al. [83]: a task is the process of working on an information need regardless of other constraints. Therefore, the tasks in this definition are basically logical sessions. Similarly, Aswadallah et al. [12] present topically-coherent sessions, which combine a set of related information needs in an overarching task. Search tasks and complex search tasks are also defined. Where search tasks represent the atomical information need resulting in one or more queries, complex search tasks represent the overarching concept combining these atomical units. These definitions are similar to those presented by Jones and Klinkner [120]. An interesting additional distinction is made by MacKay and Watters [162]. The authors differentiate between transient and persistent tasks; transient tasks may span multiple sessions, but have a definable time period in which they are worked upon, whereas persistent tasks have an indefinite temporal scope, as the underlying behaviour is repeated again and again.

Viewing the identification of logical sessions with these definitions in mind, there is clearly a certain hierarchy. There are different levels of aggregation: from single events to queries to sessions to tasks to missions. The choice of the right level of information is a focal point. Liao et al. [145] state, while a lot of research is being done at the session or query levels, research at the task level is lacking even though the latter helps in determining user satisfaction and other use cases. The authors describe a three-level hierarchy in their work:

1. query trails, where every query is followed by a sequence of browsing, representing an atomical query session
2. session trails, representing consecutive logs detected by a temporal cut; often contain multiple information needs regardless of time constraint
3. task trails, which represent potential interleaved logs related to the same task, constrained by the session trail

After comparing the three levels by applying different use cases, they report that tasks are much more precise when determining, for example, user satisfaction. Concentrating on task trails improves an application's performance from a user's point of view. Their definitions reflect the inherent hierarchy of information-seeking episodes; user interactions have an immediate goal as well as an overarching task, resulting in a complex relationship of dependencies among them [19].

Detecting or identifying sessions is a data-driven subject, which needs to be adjusted accordingly [177]. Often, research simply uses mechanical sessions as the unit of measurement, although tasks and logical sessions are becoming more important over time [181]. Their importance and value is increasingly recognized for various use cases [69], although many studies still rely almost exclusively on mechanical boundaries. Some systems may handle data without any partitioning at all, although a reasonable choice of segmentation might boost their performance [254]. The theoretical foundation is often simply not reflected in the session definitions.

Before the different session concepts are defined more precisely in Section 4.1, for now, a working definition is needed. In general, this dissertation follows the definition from [83]. In the following sections, sessions are regarded as a simple grouping of interactions in relation to a specific information need. The boundaries and structure are defined by the respective subtype of session.

2.3 Modelling Sessions and Task Hierarchies

The following section provides a comprehensive overview of session-identification approaches. For simplicity, the different concepts are roughly classified into the aforementioned mechanical sessions and logical sessions. Logical sessions are also referred to as tasks. First, all approaches utilizing mechanical boundaries are introduced. Afterwards, the vaguer notions of logical sessions and tasks using more complex approaches are presented.

Before taking a deep dive, some surveys of the relevant literature should be mentioned. Gayo-Avello [80] provides a comprehensive introduction into this field of research. The survey presents an overview of definitions, their development over time and several session-detection methods. The author distinguishes between temporal clues, lexical clues, machine learning-based methods and heuristic-based methods. Fatima et al. [75] provide a thorough and more recent overview as well, discussing the state of research on web usage mining and session identification. The authors analyse results from 42 papers, which are classified into the categories time-oriented, structure-oriented, link-based and hybrid approaches. After discussing the results, the state of research is declared to be rather immature. All variants have limitations and advantages as well, but overall the limitations are too strong and may deeply affect all later stages of data processing or analysis if not handled with care. The same argument is also the hypothesis of this dissertation.

Many factors may be considered to model sessions. Ye and Wilson [289] investigate such factors using a qualitative approach. They present a user-defined taxonomy of six factors based on 847 real web sessions. The authors discuss multiple problems regarding factors that start or end a session, may divide sessions or connect seemingly separate sessions. They interviewed the users, who were supposed to segment their own logs with no given definition of the term session according to their own understanding. After reviewing, they come up with the following elements:

- topic change (main intention change, hierarchical changes)
- task change (specific tasks related to the topic)
- different phases (sequentially dependent phases, i.e. looking for something and ordering are two different phases)
- different people (communities change behaviour)
- time gap (traditional measure)
- multitasking (diverse activity)

Each of these factors might affect the logical segmentation of a log according to actual user behaviour. They must be considered in combination to get an accurate session representation in practice; there are no specific trigger events. The resulting taxonomy of factors summarizes the foundation for how to model user behaviour, not only for queries and search engines but for all kinds of information systems. There are inherent factors that are not controllable with only log data at hand, but many are interpretable or measurable by simply looking at the data. For example, the interaction log does not reveal anything about communities; there is no visible evidence of users changing behaviour in different communities – that is, either in different contexts or when with other people. Another example would be unexpected events. If a user stops mid-task because s/he needs to attend to other errands, the data would indicate only so much. By modelling sessions, researchers aim to get a close-as-possible estimation of the user's behaviour. There are different ways to approach a reasonable segmentation by taking different features into account. These are outlined in the following sections.

2.3.1 Time-Based Approaches

An inactivity timeout is the classic, traditional way of identifying sessions. It was not only the first way to segment interaction logs, but it is also the most easy and efficient.

Time constraint	Publication
5 minutes	[107, 232, 258]
10 minutes	[82, 247]
15 minutes	[5, 41, 61, 82, 95, 96, 292, 296]
20 minutes	[30, 63, 95, 148, 189, 236]
25.5 minutes	[43]
30 minutes	[10, 12, 20, 23, 28, 35, 36, 37, 47, 48, 54, 65, 104, 120, 123, 137, 140, 143, 149, 156, 160, 173, 178, 190, 194, 198, 201, 202, 210, 217, 218, 237, 247, 248, 261, 263, 274, 286, 295]
60 minutes	[91, 234, 267]
24 hours	[269]

Table 2.1: Use of inactivity timeouts in the reviewed literature. Note: Some of these studies deal exclusively with session identification, others deal with the application of interaction data.

Up to this day, the well-known 30-minute inactivity timeout is the industry standard in many applications [23, 62], including, for example, widespread web analytics software such as Google Analytics[1]. The assumption behind temporal constraints is that proximity in time is likely when dealing with any kind of information-seeking process. Another reason to opt for an inactivity time limit might be the easy implementation compared to other approaches [89]. This is especially true for any kind of web application aiming to support users in real time. Table 2.1 shows an extract of the distribution of used inactivity timeouts in the literature observed while working on this dissertation.

The 30-minute inactivity timeout is very much prevalent. It is the classical time constraint, probably evolved from the value proposed by Catledge and Pitkow [43]. This study is among the first to introduce a temporal constraint. During their analysis on client-side tracked behaviour logs, the authors report an average time of 9.3 minutes between interactions. Adding 1.5 standard deviations, they ended up with a temporal inactivity constraint of 25.5 minutes, which introduced a wave of studies. These 25.5 minutes have likely evolved into the aforementioned 30-minute inactivity timeout, which is still used to this day.

Other research took up on that timeout. With the log data preprocessor (LODAP), Castellano et al. [42] present software for automatically extracting sessions from log files. Their tool preprocesses log files and divides requests by using a time-based constraint approach for identifying user sessions: 30 minutes as a maximum time gap and two seconds as a minimum time gap between interactions. Boldi et al. [28] introduce a graph concept for modelling behavioural patterns and query dependencies of users based on sessions built with a 30-minute inactivity timeout, although the actual concept of query chains transcends the session definition by involving all related queries by all users.

Some studies directly question the global 30-minute timeout. Halfaker et al. [91] define sessions as a short period of contiguous time involving user activity of any kind. They argue that global inactivity thresholds may be appropriate, considering this definition. To prove this, they plot a histogram over logarithmically scaled time gaps between user interactions to find valleys of inactivity. Afterwards, a two-component Gaussian Mixture model is applied to the dataset using expectation maximization. The approach is tested on

[1] `https://support.google.com/analytics/answer/2731565`, retrieved 8 June 2020.

seven different systems with multiple interaction mechanisms, resulting in 12 datasets. The results indicate that setting a global inactivity timeout of 60 minutes should be appropriate for most logs, as there seems to be a natural valley of inactivity between one minute and one day, centred on 60 minutes. Later, similar experiments are reproduced by Mehrotra et al. [178] on datasets of digital voice assistants. They find that common identification methods from prior works dealing with search engines are not applicable. Confirming the results from Halfaker et al. [91], they find that their experiments support the idea of a 30-minute timeout for search engines and an optimal timeout of two minutes for the voice assistant data.

Other timeouts have been proposed as well. Silverstein et al. [232] intended to capture all involvement by a user focusing on one single information need. Although acknowledging that this might not be representative for real usage in general, they employed an inactivity timeout of five minutes. After analysing a dataset from the AltaVista search engine with roughly a billion rows, one of their observations was that 63% of all observed sessions consist of only one request, but with an average of 2.02 queries per session. Bearing in mind the five-minute inactivity protocol, this might explain the number of short sessions – the five-minute timeout leading to shorter sessions in general.

He and Göker [95] experimented with two data sets from Excite and Reuters to find a session timeout interval that fits most applications. They tested with different limits from one minute to 200 minutes to look at the changes in distribution of different activities per session definition. They aimed to get sessions with a relatively small number of interactions, as previous (qualitative) studies regarding query behaviour reported smaller numbers of interactions. Despite acknowledging possible error sources, such as not considering semantic relations between different segments, their results indicated an optimal session interval timeout to be somewhere between 10 and 15 minutes. The same timeout period is reported by Göker and He [82] again, whose aim was to group activities belonging to a user acting in a specific role, that is, wanting to fulfil a specific information need. With the goal to prove whether time is a reliable enough marker for detecting session boundaries, they tried different time intervals as inactivity timeouts between interactions and looked at the distributions of interactions per defined session to see if there is a pattern.

Other works question the use of a fixed timeout at all. Wolfram et al. [281, 282] analysed three interaction logs – from an academic website, a general search engine log and data from a consumer health-information website – to dive deeper into user behaviour clustering. They used the so-called 80/20 rule, whereby they used the time associated with the 80th percentile of inter-query intervals as an inactivity cut for the data. This method results in optimal timeouts as 11.8 minutes for the academic website, 17.9 minutes for the search engine and 3.8 minutes for the public health website. Kapusta et al. [125] tested different thresholds on a bank portal, differentiating between content and navigational pages to determine a reasonable time spent on site for the respective type of page. They used this to model session timeout values. Their results indicate that it might be enough to just use the average time spent on site as a threshold. Similar insights are reported by Cooley et al. [55] and Spiliopoulou and Faulstich [239].

Similarly, Yuankang and Zhiqiu [291] use a dynamically calculated threshold. In the first step, they create a website topology in which they define frame and subframe pages. Afterwards, the threshold per web page is calculated via access time depending on the content of the page. He and Wang [97] present a dynamically calculated approach as well. The authors calculate a degree of importance for every web page in combination with the average time spent on the individual web page, combining these values into a relative threshold. When calculating sessions, the timeout values are altered depending on the web page. A similar approach is proposed by Dinuca and Ciobanu [64]. The authors solely rely on the average visit duration on specific pages in the data.

Mehrzadi and Feitelson [182] argue that a global threshold would create artefacts in the data and therefore taint any research relying on it. They propose an algorithm that calculates a personal timeout threshold on a user basis. The method puts the inter-query times between the queries of a user in logarithmically-sized bins. Each bin gets a score based on the two maximums on either side, resulting in a highest-scoring bin that represents the threshold. Compared with human annotated sessions, their boundary delivers relatively consistent results but often sets lower thresholds.

Peng and Zhao [207] argue that mechanical thresholds do not account for different user behaviours. When not considering different behaviour patterns, sessions may be falsely connected or cut. To evade these errors, the authors propose a dynamically calculated algorithm with a two-step average threshold. In the first iteration, the average threshold is calculated. In the second processing step, any errors are removed by checking identified sessions again.

Another direction of time-based approaches uses a fixed time interval instead of an inactivity timeout: a starting event is defined to which all following events in a fixed period of time are connected. This method is even more restricted than the inactivity timeout and therefore relatively rarely used. Nottorf [196] uses such a manually defined time interval as the session boundary. A session is a sequence of interactions that do not exceed the period of 60 minutes. With this data, a Bayesian mixture for modelling consumer clickstreams is applied. A similar segmentation approach is done by Boughareb and Farah [29], where sessions are limited to one hour to model information needs.

As Gomes et al. state, 'one of the difficulties in using a global temporal threshold is that true session intervals usually have a smooth distribution, and it is almost guaranteed that longer sessions will be handled incorrectly' [83, p. 186]. Actually, this works in both directions, as shorter segments might also easily be bundled together. This is an inherent problem of time-based approaches, as they only connect events in a period without acknowledging the content. The more advanced methods, looking at distributions, work for connecting close and potentially related events but they also can only do so much in reflecting plain behaviour, not the need behind it. Opinions on temporal approaches differ. Multiple studies report that choice of timeout is arbitrary and does not have significant impact on distributions [33, 146], others report contrary results. Many works recognize the need for other clues though. Another direction is reflected in content-based approaches, where the content and structure of the page is utilized.

2.3.2 Content-Based Approaches

Theoretically, identification methods should consider every information source the user has accessed at the time of interaction. This includes the content of a viewed page; information about the contents displayed, and most likely consumed by the user, supports determining the type of user or predicting the outcome of a sequence of interactions [1, 168]. Content-based approaches aim to identify sessions by either using the contents of a visited page or by closely reconstructing the user's path. In fact, the former can be used to construct logical sessions, as the contents of a visited page may be a good indicator of task boundaries. The latter tries to capture all interactions of the user during one visit.

One of the earliest (and simplest) methods for content-based session identification is the maximal forward reference proposed by Chen et al. [45] in 1998. According to them, a session represents the sequence of all events from a starting event until a backward reference is made. A backward reference refers to an already-visited page. This method, while rather simplistic and limited, may nonetheless be able to capture atomical information needs, depending on the system.

In 2003, Spiliopoulou et al. [238] carried out fundamental research on an evaluation framework able to compare different approaches to session modelling. They presented strategies for comparing different approaches of reconstructing activity of users from log files. Three approaches were tested: inactivity timeout, fixed length and a referral-based method. The latter only connects events to sessions if the target event is connectable via an HTTP referer to the previous URL – if they match, a path can be created to reproduce the clickstream of the user. Padala et al. [200] combine a referral-based method with the usual 30-minute inactivity timeout to identify sessions. Similar concepts are used by Pratap et al. [211] and Bayir et al. [17]. Likewise, Jiang et al. [117] propose a combined concept as well. They employ inter-activity time of users along with a Gaussian distribution to model a dynamic threshold as well as combine these with a referer-based strategy.

The same approach is used again by Chitraa and Thanamani [51], where the authors state that sessions 'can be defined as a set of pages visited by the same user within the duration of one particular visit to a website' [51, p. 24]. For the identification, navigational patterns are used. Interestingly, they state that any temporal constraint is not reliable; the user may get involved in other activities, or technical factors like loading time are neglected. Movement through the system is also considered – every subsequent request has to be connected to the prior one by a referring URL. If there is no connection, a new session is created. In their proposed algorithm, the authors try to identify relevant requests by assigning weights in relation to the time spent browsing on a page. This may help to reduce the amount of processed traffic and to identify the user's needs more easily.

Meiss et al. [184] analyse the interaction logs from 1,000 undergraduate students over the course of two months. They show that time-based segmentation approaches are not precise enough. They initially apply different inactivity timeouts and examine their effect on different measures. The results show that there are strong dependencies from all measures (like mean number of sessions per user, session duration or number of requests) to the chosen timeout, indicating that it is completely arbitrary. There are no regularities in inter-activity times. The approach they present therefore moves away from this assumption and utilizes referer trees instead – using referring URL and target URL in a request, they are able to precisely model the clickstream of a user.

These approaches all understand the term session as a single visit to a website or the single interaction with an information system. Considering this, methods based on the behaviour path of a user through a website are relatively close to the mechanical definition of a session, as they represent a completed visit. However, these approaches are not necessarily constructing mechanical sessions, as the identification based on browsing behaviour lacks the mechanical totality of the time-based approaches. Technically, these types of session cannot be regarded as either mechanical or logical, as they are simply replicating user behaviour on the system during one visit. Still, the modelling of those visits does not necessarily make for an accurate representation of the process of working on an information need, as it disregards any specifics of user behaviour. Other approaches utilize the actual content of a page instead, in order to get an estimation of the behavioural process.

Chitra and Kalpana [50] propose a graph-based approach. Each visited web page represents a node with calculated weights while the connections simulate the navigational paths. The weights represent the degree of importance per web page. A very similar approach is used by Heydari et al. [100]. The authors combine a graph representation of web pages with a statistical analysis of browsing time. The website topology is represented as weighted vertices, where the weight of each node is calculated via browsing time. This approach is similar to the one proposed by Menasalvas et al. [185], who present the concept of subsessions. These are smaller segments of a session based on the most frequent path taken on the website. The authors introduce a frequent-behaviour path tree, which basically calculates the frequent path based on historical data and utilizes thresholds to form sessions into smaller segments. Their algorithm does not use temporal features, only the navigational properties in the dataset.

2.3.3 Lexical Similarity

Other forms of content-based approaches use lexical similarity. These types of approach represent a direct move from purely mechanical sessions towards logical sessions. Moving away from representing individual visits to information systems using URL-/referer structures and mechanical boundaries as time constraints, lexical clues look at the similarity of queries to estimate whether they belong to the same information need. Considering this, lexical similarity aims for logical sessions.

The general idea behind lexical similarity is that queries which do not share a term with an earlier query are likely to indicate a new session. Basically, all approaches compare either successive queries according to some lexical measures or all query pairs submitted by a user. Comparing successive queries leads to atomical information units that likely tackle the same information need through basic reformulation techniques. Comparing all query pairs respectively enables the construction of overarching information needs. Many of these approaches combine the similarity between queries with an inactivity timeout, following the initial assumption that overall accuracy is increased by using temporal proximity.

One of the earliest and most influential works applying lexical patterns is provided by He et al. [96] in 2002. The authors identify different search patterns by comparing query terms, resulting in a classification with eight different patterns: browsing, generalization, specialization, reformulation, repetition, new (topic), relevance feedback and an 'other' category. They compare adjacent queries to find the correct classification for the respective search pattern. Basically, their classifier directly compares the terms between the adjacent queries to assign a category. Considering session boundaries, only 'new' would indicate the beginning of a new session. By their definition, a session is a sequence of activities related to each other through an evolving information need as well as through being close together in time. They performed extensive analysis with different time intervals combined with the aforementioned representations of search patterns. Using the Dempster-Shafer method[2], they identify sessions with significantly better results than using time intervals or the search patterns alone. Their algorithm classifies two consecutive interactions to either session shifts or session continuation, depending on if they belong to the same context.

Identical categories were already proposed by Lau and Horvitz [133]. The authors manually partition queries from an Excite search log into classes denoting the current search action. They assigned general goals to each query with tags like current events, weather, products and services or adult content. Afterwards, Bayesian Network (BN) models were applied to capture search dynamics and to make predictions about search progress. They report interesting insights, especially about the relationship of growing inter-query intervals and the likelihood of issuing queries for a new topic. They also report interleaving search tasks, although only in small numbers. A similar study was conducted by Spink et al. [242], who also manually annotated an Excite search log to find reformulation patterns. Jansen et al. [113] also use these reformulation classifications. They embed the concept of sessions into a broader context, which they call search episodes. They define these episodes 'as a temporal series of interactions among a searcher, a Web system, and the content provided by that system within a specific period' [113, p. 862], which is basically a very generic session definition. They assume that success or failure on session level is the critical determinant in the user's perception of the system's performance. Another assumption is that varying search patterns may announce the start of a new session.

Also based on search patterns is the work of Jansen et al. [114]. The authors classify query reformulation patterns to identify new session boundaries using the same categories again. Their sessions always begin with an initial query – which is the first query by a user on a day. The queries that follow that do not share any terms with the previous query are also considered as the start of a new session. They conduct experiments with the algorithm from He et al. [96] and compare the outcome to multiple baselines, including a 30-minute

[2] The theory of belief functions, a mathematical theorem on probabilities and reasoning with uncertainty.

inactivity timeout. Their findings indicate that users utilize system feedback and retrieved terms to refine their query according to their information need.

Detecting and classifying reformulation patterns to detect sessions is a relatively easy but also simplistic method. He et al. [96] state that by keeping the search context in mind by using reformulation patterns as the identifier, it is possible to get a grip on the larger context for the respective information need. They utilize reformulation patterns as a means of bridging the limits of the sole use of a temporal constraint by mitigating the totality of the time gap. By identifying the context, sessions can be connected even if they appear in different times. Other approaches move away from simply comparing terms to directly comparing the characters of the queries via different heuristics. This is very similar to the aforementioned variants, but can be considered more precise depending on the heuristic used. Patterns are not considered; these approaches simply calculate a score and decide session boundaries by applying a threshold to this score.

Shi and Yang [231] present a sliding window time segmentation which is supported by lexical similarity in the form of Levenshtein distance. They define a user session as the 'history of all query records that belong to the same user' [231, p. 943] in a query log. Trying to get all related queries into one session for mining related queries via association rules, a dynamic sliding window with three different time constraints is proposed: the maximum length between successive queries, the maximum interval length of a user being inactive and the maximum length of the session overall. The thresholds were arbitrarily set to be five minutes, 60 minutes and 24 hours. Afterwards, the Levenshtein distance is calculated between adjacent queries to decide if they belong to the same session.

According to Radlinski and Joachims [215], they were the first to no longer treat every query in a session individually, but to look at them sequentially. Sequences of reformulated related queries are called 'query chains'. They present an algorithm based on Support Vector Machine (SVM) classifiers to identify query chains and use these to learn preference judgements to be able to evaluate search engines with respect to reformulations. The SVM works on multiple features such as the cosine distance between query terms or the cosine distance between retrieved documents. Results show that the adaptation of query chains as a form of contextual segmentation leads to better ranking functions compared to static ones or those not considering reformulations at all.

Zhou et al. [297] analyse 47,387 queries in 18,102 sessions from 2,910 users from the Chinese website Taobao.com[3] to examine characteristics of multitasking product search sessions compared to monotasking sessions. To identify initial sessions, they use an inactivity timeout of 45 minutes. To identify tasks as the logical construct, a hierarchical clustering algorithm was applied to the pairwise Jaccard similarity between query terms with a threshold of 0.35. They defined a session as a multitasking session when the session contains queries that are matched to two or more of the product categories. They discovered that users dealt with multiple tasks in 35.7% of all examined sessions. These multitasking sessions tend to last longer than monotasking sessions, also the number of

[3] `https://world.taobao.com/`, retrieved 5 January 2022.

queries seems to be slightly higher. Furthermore, about 80% of the tasks in these multitasking sessions are unrelated while 20% are of a hierarchical or sibling like nature. While their initial session identification works by applying a temporal inactivity timeout, the use of a Jaccard coefficient creates logically related segments.

The same methodology is repeated by Zhou et al. [298] in 2016. They initially define a session as a set of queries within a certain time period meant to fulfil a single information need. Sessions are constructed with a 45-minute inactivity timeout. For the logical segments, a pairwise Jaccard index is calculated between all queries by a user and stored in a similarity matrix. Then, sequential comparison between queries decides if they belong to the same task when the similarity score is above a certain threshold. Afterwards, two hierarchical clustering algorithms are applied to the tasks based on average and maximum Jaccard values between the tasks. The authors report that 38.6% of all search sessions are multi-tasking sessions, which is much higher than the numbers reported by Spink et al. [243] (11.4%). They also report statistics comparing monotasking and multitasking sessions: apparently, the number of queries and their respective length per task (1.43 to 1.47 and 7.3 to 7.6) stays the same and users spent more time overall and less time per task in multitasking sessions. They acknowledge that their study only pays regard to query terms, which is a strong limitation especially in product searches.

In addition to an extensive literature review, Gayo-Avello [80] also tried to answer two research questions: how to best evaluate session-detection methods and what are the most appropriate identification methods. The author proposes a new method for session detection based on geometric interpretation of the time interval and the similarity between queries. Following the common assumptions that longer time intervals between queries indicate a lesser probability and that greater (lexical) similarity indicates a substantial probability of belonging to the same session, the author combines time-based and lexical distance. The actual algorithm normalizes temporal distance and lexical distance in the range [0, 1]. Temporal distance is calculated by comparing the timestamps of two queries and dividing the result by a predefined threshold for maximum session length – in this case, 24 hours. Lexical distance is calculated by comparing n-grams of queries and sessions. Each time a new query comes up, its n-gram representation will be compared to the n-grams of all previous queries in this session. This way, the method proposed gets two values in the range [0, 1] for every pair of adjacent queries, which enables the algorithm to depict the relation in a 2D-vector space. Centred on point [1, 1] in this space, the author choses a unit circle enclosing both axes to define the space which indicates the same session.

Liao et al. [144] introduce 'task trails' as a new concept, where the term task equals an atomic user information need. The authors remark that despite having seen a lot of comparable research, analysis at the session or query levels may lose information compared to the task level, as the search behaviour intertwines. Analysing a Bing interaction log with half a billion sessions, they report that 30% of their mechanically divided sessions contain multiple tasks and around 5% contain interleaved tasks. In their study, they compare the impact of session trails, query trails and task trails among multiple use cases: determining user satisfaction, predicting user interests and query suggestions.

Sessions were segmented via the common inactivity timeout of 30 minutes. To identify task trails, they first calculated the similarity between all query pairs within a mechanical session by utilizing temporal and lexical features: temporal difference between queries, Levenshtein distance before and after removing stop words, average rate of common terms, rate of common characters, length of longest common substring and more. The features were weighted with a linear SVM. Afterwards, a clustering algorithm, which they call QTC (for Query Task Clustering), groups within-session queries into tasks, by modelling an undirected graph structure and extracting all connected components as tasks, while pruning weak edges where the similarity score falls below a threshold. The results of their experiments are highly interesting:

1 determining user satisfaction based on query- or session-level parameters is not as precise as basing it on task-level values

2 tasks are able to preserve topic similarity between query pairs

3 query-suggestion based on tasks delivers complementary results to other models

Although limiting themselves to within-session tasks, these findings imply potential for more exploration. Systems can benefit from the additional sensitivity that task trails add when trying to estimate user satisfaction.

Piwowarski et al. [209] aim to identify patterns of varying search behaviours. They use the definition of query chains introduced by Radlinski and Joachims [215] to group queries belonging to the same information need. To achieve this, a tree-based layered BN is used to analyse the constructed query chains. Experiments were conducted on a proprietary interaction log from a commercial search engine, containing 57 days of data. The authors' first step was to identify mechanical sessions using the common inactivity timeout of 30 minutes, resulting in 65 million sessions. Based on this, they built query chains in three steps: concatenating all atomic sessions into one sequence, segmenting this with a global time threshold based on average inter-event time and further segmenting this by calculating a threshold based on the lexical similarity of adjacent queries inside the smaller segments. Lexical similarity is calculated based on the character n-grams of the query-pair strings:

- cosine distance between the two vectors of character n-gram frequencies
- degree of inclusion between reference query and successor
- degree of inclusion between successor and reference query

The degree of inclusion represents the probability that a query's character n-grams appear in the respective counterpart. Analysing the new query chains, they report 1.2 queries per chain with a standard deviation of 0.6, ending up with 19,196,791 query chains in total. These chains are only able to connect adjacent tasks related to the same information need, as there is no interleaving involved. Afterwards, the authors use their layered BN to develop a hierarchy of interactions of the user, similar to using implicit feedback. The network basically keeps track of the states of discrete latent variables as a summary, that is, it associates every (implicit) variable below a certain level to this level. Using a classifier, they used the generated latent variables to assess relevance for documents to successfully validate their model. Basically, their results indicate that predicting user satisfaction should not happen at event level. The whole search process reflects satisfaction much more accurately than focusing on single queries.

Depending on the definition, lexical similarity might be a good and easily implemented tool to divide logs into logically related segments. Especially when considering reformulations, complementary information like the Levenshtein distance or Jaccard indices are powerful tools. When trying to determine the task or actual information need behind user activity, simple lexical similarity should not be the only criteria [166]. Lexical similarity basically only compares the strings or characters of a query. Relying solely on superficial text features might lead to mismatches. Some queries can be considered difficult, as they are linguistically ambiguous and might not share any terms or lexical features [191]. Cross-session tasks or complex search tasks often suffer from this problem, as they typically correspond to an overarching information need which might not be reflected in the queries [131]. To elude the limitations of purely textual features, some works proposed aiming for the comparison of query meanings instead of text similarity. Semantic similarity is slightly more complex, but allows a much more precise connection of related interactions.

2.3.4 Semantic Similarity

The aim of Semantic similarity is not to achieve a direct comparison between queries but to attain a contextual comparison. A classic example of semantic similarity comes from Huang et al. [108], who apply language modelling to the task of session detection. They treat a session as a sequence, using language modelling to estimate the probability of an event following a sequence. Instead of word and term sequences, user interactions are employed. They argue that, similar to natural language, when the entropy changes (up to a certain threshold) in a sequence, a boundary can be set as the events are likely to be associated with a new topic. They utilize the calculated uncertainty of an event occurring. For the calculation of the entropy, n-gram language modelling is used. This method estimates the probability of an event by considering n preceding events. They test multiple n-gram models with varying performance. The results are promising, although very sensitive to hyperparameters and the number of n. The same approach is extended and successfully applied to database logs by Yao et al. [288]. Following up on the general idea of this method, Chierichetti et al. [49] debate whether web users actually follow a Markovian model in reality. The Markovian model assumes that a user's visit on a web page is only influenced by the previous page, any pages beforehand are not considered. The authors suggest that variable-length Markov chains are preferable to traditional or higher-order chains

Other approaches do not utilize the recent history of the user, but instead enrich the reference query with external information. In most cases, these approaches still rely on lexical similarity. The difference is the conceptual level brought by utilizing external information to achieve a broader context for any query. Daoud et al. [58, 59] build query and user profiles based on the Open Directory Project (ODP) ontology[4]. Sessions are then detected by measuring shifts in the concept extracted from vectors in the ontology. A conceptual correlation degree is calculated using the Kendall correlation measure between the user profile up to a reference query and the new reference query, resulting in conceptually related segments.

Li et al. [139] present a generative model utilizing topic membership and topic transition probabilities to segment interaction logs in search tasks and search missions. They propose a generalized hidden semi-Markov model that estimates the membership of queries topic with variational inference based on content and search behaviour. Using transition probabilities, effectively it is able to determine search task hierarchies based on the so-called search factors latent in the behavioural features.

Another variant is shown by Hua et al. [106]. Following the findings of Liao et al. [144], their aim is to identify tasks to segment interaction logs using a combination of features. They employ lexical features like Jaccard distance, temporal distance between queries and conceptual features by obtaining concept clusters using the Probase knowledge database[5] to calculate the similarity between query pairs. Afterwards, tasks are identified by implementing a graph-based algorithm which they call SCM (Sequential Cut and Merge). The algorithm takes predefined sessions as an input, builds query chains and calculates the similarity between consecutive queries connected by an edge. If the similarity is above a certain threshold, the connected queries are merged into a subtask (in a bag-of-words representation). The subtasks are then modelled in another graph structure, and the similarity between them is calculated and again merged if the similarity is above a certain threshold.

Hienert and Kern [103] use information from a thesaurus and a digital library itself to model logical sessions. By mining interactions, they use queries and viewed documents to extract keywords from a thesaurus, then query a lookup table from the digital library to get related categories for these keywords. By weighting the categories, they are able to retain the most important one, representing a topic for each interaction. They segment sessions topically by comparing subject areas and checking for Levenshtein distance between adjacent queries. Evaluation was carried out manually by assessing logs from a specialized digital library for social sciences with 100 sessions, although it is not clear how these were initially segmented.

Zhang et al. [293] present a method for intent representations, employing a recurrent architecture that learns query embeddings. These embeddings use implicit feedback in the form of shared clicks on retrieved documents, therefore circumventing classical bag-of-words problems like identifying lexically distant but semantically related terms. Using

[4] No longer accessible. Archived version available on `https://dmoz-odp.org/`, retrieved 8 December 2021.

[5] `https://www.microsoft.com/en-us/research/project/probase/`, retrieved 9 November 2021.

these embeddings, it would be easily possible to group longer user tasks, although in their study only the quality of representations was evaluated. For analysing search sessions, they used the common 30-minute inactivity timeout. Somewhat similar to this is the proposal by Singh et al. [233], where clickstream representations were used to construct unified product embeddings, for example, in recommendations about similar products. However, it is not clear how the input sessions were constructed. Also using embeddings, Völske et al. [262] utilized the so-called Word Mover's Distance (WMD) described by Kusner et al. [132]. The WMD is basically the distance between the embeddings of two strings in the vector space.

Wang and Lu [269] propose a complex search task model (CSTM), a model for grouping queries into tasks and subtasks. The algorithm effectively extracts labels for different subtasks to form the hierarchy of a complex search task. CSTM utilizes query features (i.e. query relationships based on sessions with a 24-hour inactivity split) and external resources aimed at the joint appearance of queries: taking data from search engine result pages (SERPs) and from community question answering systems. A task-coherence measure is calculated to form clusters, afterwards a Latent Dirichlet Allocation (LDA) model extracts subtask goals. Their results indicate good performance, although the authors note multiple problems. For one, subtasks could belong to different complex tasks. On the other hand, some subtasks have to be performed sequentially, causing problems in recommending tasks. They also acknowledge multilevel hierarchies of tasks instead of the simplified two-level architecture.

Contrary to earlier methods, Ustinovskiy et al. [254] deal not only with queries but also with browsing behaviour to identify logical sessions. Their algorithm uses 29 features to decide whether two pages visited by a user belong to the same session. These include URL features (i.e. cosine distance from tri-grams, length of the longest common substring), textual features (using the cosine distance between terms extracted from the pages) and temporal distances (time interval, pages visited in-between) on all pairs of pages where one of the pages precedes the other. Afterwards, they model the segmentation task of the classifier output as an optimization problem. They aim for the maximum joint probability of the partition, proposing several algorithms to tackle the calculation problem.

Aswadallah et al. [12] work on building complex search tasks based on sessions as well. In their study, they define sessions with the 30-minute inactivity timeout and construct multiple definitions from this. A topically-coherent session consists of only one or at least related information needs within the same overarching session. A search task refers to an atomical information need that again may result in multiple queries. 'A complex search task is a multi-aspect or a multi-step information need consisting of a set of related tasks' [12, p. 831]. Their algorithm is basically an association graph that uses entity representations of queries to extract and connect multiple tasks based on dependency rules. This results in high precision in terms of tasks but has its drawbacks on recall. Also, their preprocessing limits the outcome. Similar studies have been presented by Verma and Yilmaz [259, 260].

Jones and Klinkner [120] take a deep dive into the hierarchical relationships between queries belonging to the same information need. Their work is among the most influential in this field of research, as they are among the first to acknowledge the probability of multiple tasks related to a single information need. They analyse the effect of typical timeouts used for segmenting logs into sessions, perform a hierarchical analysis of search tasks to identify short- and longer-term goals as well as compare previously published approaches that have been used to identify search tasks. As a starting point, they distinguish between search sessions, search goals and search missions. Sessions here are just a basic physical unit – the number of interactions by a user within a fixed time window. There is no information need involved as in other definitions, as the needs are defined via goals and missions. Search goals is the atomic component, representing a single information need reflected in one or more queries. The missions are the overarching concept, connecting related information needs together from one or more goals.

For their experiments, they used 312 user sessions from Yahoo over the course of three days. The data was extensively manually annotated to reflect search goals and search missions for every query. After annotation, they had 312 user sessions with 1,820 missions, 2,922 goals and 8,226 queries. Descriptive statistics revealed, for example, that 63% of the annotated search goals were tackled within one minute, but 15% spanned time periods longer than 30 minutes. It is also reported that 16% of goals and 17% of missions are revisited or interleaved with other goals and missions. They also tested different timeout values in their data – from assuming every interaction can be considered separately to calculated thresholds (five and 13 minutes) to longer periods ranging from 30 to 120 minutes. The results indicate that the choice of timeout can be arbitrary, as different timeouts give similar accuracy regarding the goals and missions. The interleaving nature of queries within the evolving process of pursuing an information need must be considered, otherwise tasks will be disconnected from one another. They contribute the automatic mapping of queries to search goals and missions. Assuming goals and missions are pursued interleavingly, every single query has to be compared with one another. This results in 305,946 queries. They formulated this as a classification problem, for which they used a logistical regression with 10-fold cross-validation on the dataset. They used features from four different areas:

- temporal (inter-query time, adjacency)
- lexical (Jaccard distance between sets of words, Levenshtein edit distance)
- co-occurrence (using a bigger sample of query pairs to identify the likelihood of them occurring together)
- semantic (distance between documents retrieved from both queries of a pair)

The authors found they received the best results when features from all areas were included compared to testing with different sets of combinations, reaching an accuracy of 89% in all four tasks (mission / goal boundary, same mission / goal). Used alone, the lexical measures performed best. They also found it easier to detect boundaries between queries than matching pairs to the same goal or mission. The results indicate that their hierarchical model may help to identify whether a user is on a simple or more complex task. With the inclusion of boundaries, it was possible to evaluate the complexity of a task.

Lucchese et al. [154] deem the well-known time-based limits for sessions unsuitable when trying to identify tasks because they a) break potential longer information needs and b) mix multiple and interleaving needs. The authors focus on identifying task-based sessions by estimating the similarity of query pairs, following the general idea of Jones and Klinkner [120]. For their theoretical model, they differentiate between mechanical sessions with an inactivity timeout of 26 minutes (after measuring the distribution of time gaps between all queries in the log) and task-based logical sessions, for which they acknowledge the existence of different interleaving needs. This means that queries belonging to the same need are not necessarily consecutive. The experiments are based on a 2006 AOL search engine query log, consisting of roughly 20 million queries by 657,000 users over the course of three months.

As a first step, the authors created a ground-truth dataset by manually annotating a sample of 2,004 queries, ending up with a total of 1,424 queries in 307 sessions after processing. In these mechanical sessions, there were 554 task-based sessions with 2.56 queries on average per task. Furthermore, 50% of the time-gap sessions contained only a single task, with an average of 1.8 tasks per session. But, interestingly, 74% of web queries were embedded in multiple tasks. Afterwards, they experimented with four different clustering algorithms and compared the results to the ground-truth (using k-means clustering, HDBSCAN and graph-based variants).

Feature-wise, lexical features like Jaccard indices on trigrams and Levenshtein distance are used as well as semantic-based features using external knowledge sources (Wikipedia and Wiktionary). The algorithm computes distances between query pairs that are then used for clustering. As it stands, utilizing external sources like Wikipedia to understand semantic relationships between queries seems to be very beneficial in terms of constructing task-based sessions. The same algorithm is used by Feild and Allan [76] to decide if two queries belong to the same task. In this study, the authors aim for query recommendations based on task relatedness to improve user support.

Hagen et al. [89] propose a cascading method to connect consecutive queries with the same information need. They also try to quantify the relationship between runtime improvements and accuracy measures. Their general assumption is that there are different categories that grow in complexity when considering session boundaries, so these different categories should be checked sequentially. Checking first for lexical and afterwards for semantic conceptual similarity follows that premise. The authors perform the following steps cascadingly: 1. query comparison with lexical similarity, 2. geometric approach from [80], 3. Explicit Semantic Analysis (ESA) using tf/idf and Wikipedia articles and 4. comparing shared search results. The same approach is then revisited in 2013 [88]. They construct search missions using an improved version of the cascading approach. Their strategy consists of five steps: time-based with a 90-minute timeout, simple patterns (keywords), lexical similarity and time, using Wikipedia as an external index with ESA (based on research by Gabrilovich and Markovitch [79]) and using Linked Open Data (LOD) as an external index, comparing the first 10 search results. The actual improvements can be found in extending the geometric method and using LOD as another level of semantic similarity.

Built on the approaches presented by Gayo-Avello [80] and Hagen et al. [88, 89], Gomes et al. [83] use the same method, but add an additional layer if a reliable decision cannot be made. Their algorithm consists of three steps: using a per-user threshold for temporal limits, Jaccard similarity coefficient and character 3- and 4-grams, they merge queries if the results are above certain thresholds depending on the type of feature. If the decision cannot be reliably made, they adopt the cascading approach from Hagen et al. [88, 89], and use pretrained FastText embeddings[6] to measure semantic similarity between queries and sessions in several cascading steps. The last step, if a decision is still unreliable, clicked URLs are considered using lexical distance between them. Experiments were made on the same annotated dataset used by Gayo-Avello [80]. For evaluation, they again followed Gayo-Avello using precision and recall in addition to an F1-score. The results indicate improvements over the compared baselines.

Kotov et al. [131] combine multiple sessions with regard to the underlying information with the aim of representing cross-session tasks. Their reasoning is that information needs may span multiple sessions, which makes it harder for search systems to support the user. The article focuses on two specific problems. In the first task, the goal is to identify all queries from previous search sessions belonging to the same task – here, this task is referred to what they call an early-dominant task. Early-dominant means in this case that the queries belong to the 'first task that spans at least k distinct queries' [131, p. 8] during the first two days of a user log with the same task label in the query stream of a user. The query stream is the sequence of all the user's queries. The second problem is focused on task continuation. Given an early-dominant task label and the last query belonging to this task, the goal is to predict if the user returns to this task with future queries.

The experiments were conducted on an anonymized query log generated by a browser plugin over the course of one week. Sessions are identified as sequences of queries using a 30-minute inactivity timeout. Only users with at least five search sessions and at least 10 queries overall are analysed. Secondly, all queries were expanded using techniques presented by Radlinski et al. [216]. Also, pairs of queries are compared with the help of cluster techniques to ascertain the strength of association between them. The result of this step is a bag-of-words representation of each query using the similarity score between the query pairs. In the third step of the process, the query pairs are compared with two similarity measures and, if the similarity score is above a certain threshold, both are assigned to the same label. In the end, two datasets are used for further processing. To tackle the aforementioned tasks, the authors formulate each one as a binary classification problem. By using different features calculated for each query and query pair, the authors were able to successfully prove the ability to effectively model cross-session information needs.

[6] Based on the results from Bojanowski et al. [27], available on `https://github.com/facebookresearch/fastText`, retrieved 9 December 2021.

Similarly, Agichtein et al. [4] predicted search task continuation. The authors analysed a query log from Bing with 1,191 users and 28,474 queries over the course of one week to characterize intents, motivations and topics of long-running tasks and present multiple techniques for predicting if a searcher will continue working on a task within a certain time frame. Sessions were defined with the common 30-minute inactivity timeout. Following Kotov et al. [131] with the idea of early-dominant tasks, they manually annotated the log with specific labels describing the type and nature of the task, i.e. fact-finding, time sensitivity, complexity and the likelihood of the task being continued at a later time. Afterwards, they added topics to the tasks by employing a classifier utilizing external sources and analysed their task-continuation likelihood as well. With the gained knowledge in mind, they presented a classifier based on a gradient-boosted decision tree using multiple features founded on session history, search topic, user engagement and user profile history. By testing against different baselines and using feature ablation, they were able to outperform the baselines in prediction quality as well as to identify the most important features for predicting task continuation (user history, task engagement).

Another take on semantically related task identification is provided by Sen et al. [229], who use temporal-lexical similarity to construct semantic similarity. They utilize context features to embed query terms in a vector space in order to be able to identify tasks that span across multiple sessions. Their goal is to group queries by logically connected tasks instead of mechanically constructed sessions, whereby tasks are defined as 'a multi-aspect or a multi-step information need consisting of a set of related subtasks' [229, p. 283], following the definition of Aswadallah et al. [12] where sessions are a set of queries within a certain time period. To achieve the segmentation into cross-session tasks, they propose a method that utilizes an embedding technique driven by semantics as well as a completely unsupervised clustering algorithm. Basically, the authors use the confines of mechanical sessions created with a 26-minute inactivity timeout to create their word embeddings. The boundaries of a session represent the temporal context, while as a next step, in-session task clustering (performed with features such as lexical similarity between query terms and similarity between the top 1,000 retrieved results) provides further restrictions on the embedding context. The clustering algorithm to segment the query vectors is then carried out with one of the approaches presented by Lucchese et al. [154, 155], but globally over the complete set of queries. Experiments were conducted on the AOL query log with 1,424 queries, annotated manually with cross-session tasks. Compared to several unsupervised baselines using their own implementation but different embeddings and the original algorithm from Lucchese et al. [155], their tempo-lexical context significantly improves the resulting cluster quality.

Contrary to binary same-task classifications across sessions such as those by Kotov et al. [131] and Agichtein et al. [4], more recent studies have tried to extract tasks and, more importantly, hierarchical relationships between tasks. As Mehrotra and Yilmaz [180] described, while the binary same-task classifications are 'good at linking a new query to an on-going task, often these query links form long chains which result in a task cluster containing queries from many potentially different tasks' [180, p. 286].

Du et al. [69] also observe that binary same-task classification on query pairs works well for assigning queries to existing tasks, although it tends to construct never-ending tasks. To capture the evolving nature of user intents and query topics, the authors propose a long short-term memory (LSTM) neural network with an attention mechanism for session identification. The algorithm is able to segment sessions by learning which context is relevant for which query through word and character embeddings. While input comes in the form of query sequences, the output of the model produces smaller chunks of adjacent queries that belong to the same task. The authors consider these as the atomic information need of a user, which corresponds to a common definition of sessions. Afterwards, these session segments are clustered using one of the approaches presented by Lucchese et al. [154, 155]. The results of the two datasets were evaluated with promising results compared to typical session baselines. The authors argue that errors made while identifying sessions in the first step are magnified by the clustering process in the second step; therefore, these effects will also be projected into any downstream application, a statement which is also proven later in this dissertation. Also using a sequential model and comparing Du et al. as a baseline, Lugo et al. [157] report similarly promising results using a bidirectional recurrent neural network (RNN) to eventually detect search task boundaries. Their algorithm works by comparing the vector embedding representations of adjacent query pairs, even without additional query context. The same authors also present a recurrent deep clustering algorithm [158] reusing their previous segmentation technique, providing advanced and performant methodology to extract search tasks.

Wang et al. [266] aim to model semantic relationships in cross-session tasks. The need to identify the boundaries of original mechanical sessions is mitigated by that. To identify long-term search tasks, a semi-supervised clustering model based on latent structural SVM is applied. The major difference in their approach is modelling the task identification as a structural learning problem determining the dependency among queries instead of pair-wise binary classification as shown in [131]. They define sessions with the common inactivity timeout threshold of 30 minutes and search tasks as the overarching concept that may span multiple sessions. Interestingly, they note that their clustering algorithm does not optimize for a higher in-cluster similarity – this means, queries belonging to the same task do not necessarily have to be lexically similar, instead, the algorithm optimizes for what they call the best link, finding the strongest link between a reference query and the queries in a cluster. At least one query in the cluster has to have a strong relationship with the reference query. They model this assumption as a latent structural SVM, calling it bestlink SVM. They also present 'weak' supervision signals: a set of annotation rules are used to capture same-task queries, which proves to work well as a substitute compared to manual annotating, as the SVM relies on a fully annotated log. Results indicate, for example, that more than 57.2% of queries span across different sessions while 31.1% are interleaving.

Following a similar idea, Mehrotra and Yilmaz [181] introduce 'task embeddings', partly based on earlier studies [175, 176, 179, 180]. Their motivation is that mechanical sessions tend to contain multiple tasks, and that, therefore, using a task-based segmentation is more reasonable for any IR task than a simple mechanical representation. In the context of a reference search, for example, mechanical segmentation would pollute the data with non-related information. They borrow the approach proposed by Wang et al. [266], using latent structural SVMs. After extracting the tasks, the word embeddings are generated for all queries in a user query sequence, but only considering surrounding queries related to the same task. Experimenting on a proprietary dataset from a commercial search engine consisting of 24 million rows in eight million predefined search sessions, they build task embeddings with a context window size of two, resulting in four words as context per query term. They compared global embeddings (based on the documents returned by the queries), session embeddings (trained on the predefined search sessions), random (trained on randomly shuffled queries as context) as well as their own task-based embeddings. Results confirm their hypothesis – using sessions as context does indeed pollute the context of a query.

In search of the best way to understand, represent and utilize user intent from user interaction logs, Mehrotra [173] proposes a non-parametric Bayesian approach with multiple additions. In the dissertation, the author extracts search tasks – defined as the atomic information need resulting in one or more queries, following the definition of Jones and Klinkner [120] – using a latent structural SVM. Afterwards, Mehrotra presents further approaches for extracting subtasks and extracting hierarchies of search tasks and their respective subtasks. Throughout the experiments, sessions were identified with the 30-minute inactivity timeout.

Li et al. [138] view search tasks 'as a sequence of semantically related queries linked by influence' [138, p. 732]. They assume that queries which are not fulfilling an information need will trigger a new semantically related query. Task identification and labelling is then based on a probabilistic model using LDA modelling and Hawkes processes. Basically, the Hawkes processes are based on the aforementioned assumption that sequences of queries have some influence among each other depending on the information need. The LDA model in this case exploits the (temporal) co-occurrence among the user query base for latent information.

Considering the comparisons with natural language and how events in sequences affect each other on a conceptual level, using semantic similarity seems to be a good tool for estimating boundaries. By utilizing contextual information to enrich query representations, it is possible to detect broader topics which makes it fairly easy to detect logical sessions. However, a problem might be hidden in the hierarchical structure of queries, tasks and overarching information needs. Murray et al. [193] state that relying on semantics is dangerously circular as it might 'conceal persistence and recurrence of users' long-term information needs' [193, p. 2]. This is true for binary classification as well as for any clustering algorithm. Relying on semantics alone might not be enough, however, since it tends to generate never-ending sessions by aggregating new and seemingly related events to old sessions because of semantics, even though they might tend to a new topic.

2.4 Evaluating Session-Identification Approaches

Evaluation is supposed to be an integral part of any research project. Only by standardized methods it is possible to determine the quality of different approaches. The next section will present some commonly used datasets and methods in the area of session identification, as well as discuss why evaluation for session identification is a problem in itself. First, some widely used datasets are introduced. Following this, a short overview of common measures and methods to estimate the quality of the approaches is introduced. As the area is quite limited, the section focuses only on the most important works.

2.4.1 Datasets

The most common method to evaluate the accuracy and correctness of detected sessions is the comparison with manually labelled gold-standard datasets [88]. First, a gold-standard dataset is created – either by taking an existing one or manually annotating the one that is already being worked with. After that, the output of the algorithm is compared to this manually created gold standard in terms of the commonly used measures for evaluating IR systems. A few of the more commonly used datasets regarded as a gold standard are presented immediately below.

The first part of this section presents the most commonly used datasets for session identification. Often, studies in this field of research use proprietary datasets from commercial search engines, e-commerce web sites or from the logs of university websites [238]. In these cases, it is often not clear how the datasets are structured and what their properties are. While some studies try to give an exploratory overview of the data, in many cases this is not enough. It can be particularly challenging to reproduce results from these articles. When considering the implementation of use cases while using a proprietary dataset, there are gaps that make it difficult to understand the actual operating principle of the algorithm. This is particularly impactful when no information about the session identification is given at all. As an example, Mei et al. [183] tested their sequential framework on two proprietary datasets collected from search engine logs. One is segmented into all interactions from a user during the course of a day, containing 1.2 million queries and related clicks. The other contains 17,355 queries annotated by human judges with task labels, segmented into arbitrary search sessions. A more precise definition of these sessions is not given. They evaluated their set of tasks using a simple baseline with accuracy, recall and precision. The outcome is not really comparable though, as the structure of the data is not easily comprehensible. Missing information about datasets and sessions is a common problem [6, 11, 15, 22, 26, 101, 105, 130, 134, 136, 150, 152, 174, 221, 235, 251, 253, 264, 285], making it a real challenge to compare the algorithms or put them into context.

Interestingly, studies that focus solely on the identification of session concepts or session analysis often use subsets of the same publicly released datasets. These interaction logs usually originally stem from around the 2000s, first presented in the early studies using interaction logs and reused again in the following years. The datasets are almost exclusively focused on queries from search engines. One of the early providers was the Excite search engine[7], providing several studies with query interaction data from around 1997 to 2001 [111, 112, 240, 246]. Others used data from the AltaVista search engine[8] [115, 232, 245] or the Dogpile meta search engine[9] [113, 114]. The problem with these datasets is that they are comparatively small and focused on one day, or are, as of now, outdated. The size of the dataset, often limited to one day, is a problem when considering the variety of session definitions. Working on the assumption that sessions may span several days, these datasets are insufficient. Likewise, as these early studies mostly focus on the rather narrow end of session definitions, they might suffice.

Jones and Klinkner [120] tested their algorithm for hierarchical task identification on one week of logs from the Yahoo![10] search engine, as one of the authors was employed at Yahoo! during the time of research. Their actual dataset is relatively small though, consisting of a sample of sessions from 312 users over the course of three days. Other studies also worked with data from Yahoo! [67, 121, 210], although these were mostly proprietary as well.

In 2007, one of the most famous (and most used) datasets was released. The AOL dataset was publicly released by the AOL[11] search engine for academic research purposes. It contains over 30 million queries from 650,000 users taken from a three-month sample. The dataset had some serious flaws, though. As it was not properly anonymized, multiple users could be identified through personal information in the query strings [12]. This led to much criticism and discussions about the ethical component when working with this kind of data. Still, the dataset or at least samples from it, have been used time and again over the years [80, 90, 155, 262]. Gayo-Avello [80] created a gold standard for this dataset.

Hagen et al. [88, 89] used this gold standard created by Gayo-Avello. They explain that the evaluation of sessions or missions is usually done by evaluating the results of an algorithm against manually annotated logs. The authors also rightfully point out that there are only two publicly available datasets with a gold standard: the one from Gayo-Avello from 2009 and another one created by Lucchese et al. [154] in 2011. The latter is also a sample from the AOL log, consisting of 1,424 queries from 13 users, with mechanical sessions detected with a 26-minute inactivity timeout. Both gold-standard datasets are still used today, for example by Völske et al. [262] in 2019.

Völske et al. [262] test three datasets of task-based queries, defined with different characteristics: 1) a session-based dataset originating from the AOL log, 2) a dataset based on TREC queries and 3) a dataset based on WikiHow[13] queries. For 1, they utilized the already-published datasets described by Lucchese et al. [154] and Hagen et al. [88]. They took the queries already divided into search sessions / search missions and annotated them manually with task information. Identical queries were merged. Afterwards, all queries

[7] `http://www.excite.com/`, retrieved 5 January 2021.

[8] No longer accessible.

[9] `https://www.dogpile.com/`, retrieved 5 January 2021.

[10] `https://de.yahoo.com/`, retrieved 15 November 2021.

[11] `https://www.aol.de`, retrieved 15 November 2021.

[12] `https://en.wikipedia.org/wiki/AOL_search_data_leak`, retrieved 15 November 2021.

[13] `https://www.wikihow.com/`, retrieved 15 November 2021.

were issued to Google and Bing search engines to crawl search suggests, adding these as new queries to the same task. The resulting dataset consists of 41,780 queries aligned to 1,423 tasks. The dataset in 2 is scraped from the TREC session tracks 2012–2014, from the TREC task tracks 2015 and 2016 and Webis-TRC-12. Again, every query was issued to the search engines to collect new queries via the search suggests, resulting in 47,514 queries in 276 tasks. For 3, WikiHow was crawled following the method explained by Yang and Nyberg [287]. After collecting the search suggests, the resulting dataset consisted of 119,292 queries in 7,202 tasks.

A newer dataset comes from the Russian search engine Yandex[14]. This publicly available dataset[15] comprises 30 days of search activity, released as part of the Web Search Click Data workshop (WSCD 2014)[16] in 2014. It contains over 20 million queries from around five million users. It is used, for example, by Halder et al. [90].

With the shift to Interactive Information Retrieval (IIR) instead of traditional evaluation, new datasets were published as part of the TREC[17] conferences. Important are the TREC Session tracks [39] and the TREC Task tracks [124]. The Session Track ran from 2010 to 2014, focusing on the creation of test collections that allow the evaluation of actual user behaviour during a search. The goal is to make the evaluation of IR systems possible over sessions instead of the classic ad-hoc retrieval, where queries are treated individually. The datasets were created by showing users a description of a topic, a search engine and a list of 10 ranked results with the possibility for further pagination. In total, four datasets were created. The Task Track is similar, but its aims are to evaluate how well systems may understand the underlying tasks of a user's query or behaviour. These datasets consist of different tasks with a related query and possible candidate queries.

There are also newer developments of reusable datasets, although these are not widely used as of now. They also mostly focus on specific use cases rather than session detection, but may be used for that as well. Chen et al. [44], for example, present a new e-commerce dataset[18] containing a large number of search sessions. They preprocess in multiple steps including a 30-minute inactivity timeout and the removal of sessions with only one or more than 10 queries. Brost et al. [32] publish a music streaming session dataset for public use in research areas like music recommendation. The dataset contains logs with a session identifier, timestamp, contextual information, track information and interaction types. The predefined listening sessions are segmented by a 60-second inactivity timeout. Mayr and Kacem [171] publish a dataset with retrieval sessions over the course of one year.

[14] `https://yandex.com/`, retrieved 15 November 2021.

[15] `https://www.kaggle.com/c/yandex-personalized-web-search-challenge/overview`, retrieved 15 November 2021.

[16] `http://www.wsdm-conference.org/2014/accepted-workshops/`, retrieved 15 November 2021.

[17] `https://trec.nist.gov/`, retrieved 15 November 2021.

[18] `http://www.thuir.cn/tiangong-st/`, retrieved 15 November 2021.

The authors identified sessions with a 20-minute inactivity timeout, resulting in 484,449 retrieval sessions. The dataset is based on a academic search engine called Sowiport[19].

All these datasets are somewhat different from each other. As Hagen et al. [88] point out, even the reusable gold standards have several drawbacks. Regarding the one from Gayo-Avello [80], they criticize several preprocessing decisions (i.e. the removal of timestamps or the sampling quality of the subset) and, more importantly, the notion of the session. The aim of Gayo-Avello was for sessions that connect consecutive queries related to the same information need – there are no real hierarchies. More or less the same is stated about the gold standard of Lucchese et al. [154], where mechanical sessions were used to collate a dataset from a small number of users.

As a consequence of the many assumptions, there are numerous different definitions leading to completely different gold standards. These different definitions also have an impact on the actual methods for evaluation. The so-called gold standards are not necessarily gold standards, but are more like a variety of different facets of gold standards. The different definitions lead to different standards.

2.4.2 Methods

Evaluating sessions is challenging, not least because the evaluation is completely dependent on what the assumptions for the resulting sessions are. Still, gold standards are applied in many works; sometimes with more, sometimes with less deviations [120, 131, 180, 254]. A short but informative overview of the development is presented by Gayo-Avello in 2009 [80]. According to this author, the procedure was first suggested by He and Göker [95] in 2000 (although they did not apply a gold standard themselves). Gold standards are regarded as valid, although they have problematic implications.

Consequently, He et al. [96] introduced multiple measures for the comparison of session-detection algorithms to a manually labelled gold-standard dataset. They utilize Precision, Recall and the F-measure, adopting the well-known measures to their own algorithm. For the adaptation, they use their newly introduced concepts of session shift and session continuation, referring to a possible session change between consecutive interactions. Using their algorithm, they predict whether the time gap between activities contains either a shift or a continuation, which is reflected in their evaluation measures. The actual measures are calculated as follows:

- Precision
$$P = \frac{N_{shift\&correct}}{N_{shift} + N_{cont}}$$
- Recall
$$R = \frac{N_{shift\&correct}}{N_{true\_shift}}$$

[19] https://www.hbk-bs.de/einrichtungen/bibliothek/recherche/fachdatenbanken-alphabetisch/sowiport-csa-sozialwissenschaftliche-datenbanken/index.php, retrieved 1 December 2021. Sowiport itself is no longer accessible.

- F-measure

$$F_\beta = \frac{(1+\beta^2)PR}{\beta^2 P + R}$$

N_{shift} represents the number of detected session shifts, N_{cont} a detected continuation, $N_{shift\&correct}$ the number of correctly identified session shifts and N_{true_shift} the number of shifts identified by human judges. β is a control parameter to weigh up the importance of either Precision or Recall.

Precision and Recall are the basic and most frequently used measures for evaluation in IR [170]. In IR, Precision relates to the number of retrieved documents that are relevant in relation to all retrieved documents while Recall represents the number of retrieved relevant documents from all relevant documents. Transferred to the context of session identification, the formulas are adapted to the scenario. Precision is the number of time gaps between consecutive activities that contain correctly identified shifts by both judges and the algorithm in relation to the sum of all identified session shifts. Recall is the amount of correctly identified shifts by both judges and the algorithm in relation to the sum of all session shifts agreed upon by the human judges.

The F-measure considers Precision as well as Recall, weighted by β. He et al. [96] set β to 1.5 to put more weight on the recall. According to them, high recall equals a low number of falsely connected sessions, which is an outcome preferable to wrongly connecting unrelated sessions. This type of error is called a type A error – falsely splitting activities into different sessions although they belong to the same topic. The other variant is a type B error: missing the split of unrelated activities.

Gayo-Avello [80] picks up on these measures. The author evaluates the effectiveness of the tested session-identification method by using these formulas and adopting additional measures from Makhoul et al. [167]. According to Gayo-Avello [80], Makhoul et al. [167] point out that the F-measure is unbalanced as it ignores certain errors. As a result, the error rate (ERR) and slot error rate (SER) were introduced. These additional measures originate from the problem of Chinese word segmentation. Both are error measures adjusted to the context of segmentation quality by Gayo-Avello. The ERR measures the ratio of deletion (type B) and insertion errors (type A) compared to the number of actual correct shifts in sessions. SER is an adapted version of ERR, and more balanced regarding the relation of the different error types. The final formulas proposed by Gayo-Avello are set out below:

- Precision

$$P = \frac{N_{shift\&correct}}{N_{shift}}$$

- Recall

$$R = \frac{N_{shift\&correct}}{N_{true\_shift}}$$

- F-measure

$$F = 2\frac{N_{shift\&correct}}{N_{true\_shift} + N_{shift}}$$

- ERR

$$ERR = \frac{N_{true\_shift} + N_{shift} - 2N_{shift\&correct}}{N_{true\_shift} + N_{shift} - N_{shift\&correct}}$$

- SER

$$SER = \frac{N_{true\_shift} + N_{shift} - 2N_{shift\&correct}}{N_{true\_shift}}$$

The variables used are the same as originally introduced by He et al. [96]. Note that the equation for Precision is a corrected version[20] of the original proposition [96]. ERR and SER are adapted to represent the possible errors in session identification (i.e. type A and type B errors). Both ERR and SER are rarely used in other works.

Lucchese et al. [154] also compare their automatically extracted tasks to a manually labelled ground-truth. For this to work, they differentiate between predicted classes (the task-based logical sessions identified by the tested algorithms) and actual classes (those detected by human judges). For evaluation, they calculate the F-measure separately for every detected task and average the value over all tasks (by weighing the value according to the size of the task) to evaluate the algorithm. They also propose using the Rand index and the Jaccard index to calculate similarity between interaction pairs that have different tasks or classes with regards to the manually labelled ground-truth. The following formulas are used:

- F-measure

$$F_{i,j} = \frac{2 \times p_{i,j} \times r_{i,j}}{p_{i,j} + r_{i,j}}$$

- Rand index

$$R = \frac{f_{00} + f_{11}}{f_{00} + f_{01} + f_{10} + f_{11}}$$

- Jaccard index

$$J = \frac{+f_{f11}}{f_{01} + f_{10} + f_{11}}$$

Precision $p_{i,j}$ is calculated as 'the fraction of a task that consists of objects of a specified class' [154, p. 284], while recall $r_{i,j}$ represents the share of objects of a specified class that a task contains. f_{00} is the number of interaction pairs having a different class and task, f_{01} the number of pairs having a different class but the same task, f_{10} the number of pairs with the same class but a different task and f_{11} the number of pairs with the same class and the same task. Lucchese et al. [154] employ these measures to their own algorithms as well as several baselines like mechanical sessions in different temporal variations.

Hua et al. [106] developed an algorithm to extract conceptually related, interleaved tasks from session data. For evaluation, they looked at the effectiveness of various classifiers on subtask creation, and the accuracy of extracted tasks by comparing baselines with their algorithm. For the first evaluation task, error rate is used as a measure, because it can be denoted as the misclassification rate of edges between graph structures. To measure the

[20] Apparently, there was a misprint in the work of He et al. [96], later corrected by Ozmutlu and Çavdur [199].

accuracy of algorithms in the second evaluation task, the F-measure and Jaccard index are used. As they base their evaluation on a manually labelled dataset, the F-measure can easily be calculated by comparing the manually and automatically set tasks at the query level. The Jaccard index compares these as well, but at the query-pair level. All measures are then aggregated at the session level.

Agichtein et al. [4] also use Precision, Recall and the F-measure. The authors compare their binary classifier to the algorithm presented in [131] and human judges (given the first two days of the dataset) by evaluating the positive class of task continuations and the area under the curve using one week of data. They only allowed their algorithm to use two weeks of user history prior to the analysed week of data in the query log, although they tested it with different combinations of history and features.

Other works have tried approaching evaluation in other ways than a gold-standard dataset. Instead of manually labelling the sessions, He and Göker [95] analysed the number of events per session in relation to the inactivity timeout they used, aiming for sessions with a small number of activities – an assumption based on former research. Similarly, Jansen et al. [113] measured the effects of their session definitions on session length, which they related to the number of requests and session duration, and again referred to the time spent. However, these methods are outliers compared to the comparisons using a gold standard.

In another example, Mehrzadi and Feitelson [182] evaluated their method on the AOL dataset in two ways: looking at the results in isolation and comparing it to human judgement. In the first step, they plotted the number of sessions per session length in minutes, resulting in a much smoother graph, without obvious artificial breaks, in comparison to the hard, global timeouts. In a second step, which compared human-annotated sessions, they came to relatively consistent results, although their algorithm often set the boundary lower than the human judges. They also argued that evaluating a time-based boundary with time-based methods might result in a circular argument, which is why they also used n-grams between adjacent queries using heat maps. This approach reported mixed results; sometimes, a topical switch was cut, sometimes not. The authors argued, however, that detecting topical shifts might add useful information.

Another slightly different angle is given by Liao et al. [144]. Instead of comparing the results of their task extraction to manually annotated sessions, they chose to manually label the relationship of query pairs. This way, they were able to learn the importance of different query features (temporal and lexical) using an SVM. By having the manually labelled ground-truth of similarity between query pairs, they were able to plot the weight of query features, thereby enabling the estimation of the performance of their classification model. Satisfied with the performance of their classifier, they omitted to evaluate the performance of the clustering algorithm responsible for the actual grouping of query pairs into tasks, however, since that was not the focus of their paper. Instead, they applied their trails (query, sessions and tasks) to different use cases to see how they would fare in comparison. For this, they utilized several models to estimate user satisfaction and predict user interests and query suggestions.

The evaluation of session-identification methods is as challenging as finding a valid way to identify them in the first place. The process of evaluation is directly dependent on the underlying assumptions of what a session is, be it one constructed mechanically or logically. A descriptive example would be the discrepancy between sessions considered as all actions related to an individual information need (for simplicity: without interruption) compared to complex search tasks consisting of multiple such information needs. Given these two assumptions, a human judge would likely annotate different segments, which in turn would lead to comparison results that are actually rather unusable. This is what makes finding a standard way to evaluate session detection so challenging. Even the choice of baselines is also actually dependent on the basic assumption. Comparisons are unnecessary since the initial assumption is likely to be different.

Mehrzadi and Feitelson [182] already pointed to this issue in 2012. They stated that it is not yet clear if human judgement is a good way for creating a ground-truth. Different human judges might behave differently. There are countermeasures to this, but they are not applied regularly. This problem of data quality is not easily solvable, since the actual definition of a session has still not been fixed. Rac [214] described this problem from another perspective: human judges are not alone in having problems in objectively deciding on session boundaries; the user's intent in a session – that is, the nature of intent being pursued – might lead to completely different boundaries.

These types of data quality issues can also be found directly in the datasets. Considering the state of research, there is a lack of adequate data [182]. The available datasets differ from one another in impactful ways or lack other important features. Mehrzadi and Feitelson [182] criticize how timestamps often only denote the beginning of an interaction (a common problem in interaction logs) and may therefore disrupt the accuracy of the resulting picture. Another point here is that clicks following a query are often considered just an attribute of that query – there may be no individual log entry for every interaction.

The lack of standards in evaluation techniques is a persistent problem. Already in 2003, Spiliopoulou et al. report that 'although data preparation is essential for knowledge discovery, studies on the evaluation of data preparation methods are comparatively scarce' [238, p. 5]. Fatima et al. [75] reaffirm this in 2016 by reporting that the state of research in session identification (and, consequentially the evaluation of these methods) is still immature despite immense efforts.

In many cases, studies work with some data, often randomly sampled or arbitrarily preprocessed, use it for training or analysis and seem to be happy when the results indicate a well-working algorithm. There is a lack of critical review of these processes, especially considering their importance. This dissertation provides a comprehensive comparison of different session-identification approaches along with an objective way of showing differences.

2.5 Using Session Data in Machine Learning

This section provides only a brief overview of potential use cases for applying the concept of sessions in a production environment since this is not the main focus of the dissertation. The section is intended to give an idea of what is possible using different segmentation and how these use cases may benefit from different techniques. The section is also intended to reiterate the fact that many of these use cases apply session-identification approaches without sufficient forethought, as indicated in [62].

Overall, there is a great variety of possible applications following different directions of research and commercial areas. Roughly following the topic classification of Jiang et al. [116], three general areas are covered: recommendation and personalization, query support and user analysis. These areas either try to help users get what they want or improve a system's performance, be it commercially or with regards to usability and user experience. Recommendation and personalization are classic fields of application. Query support is about helping the user in their search by supporting them with suggestions or utilizing the user's historical interactions to improve the system's retrieval performance. Common examples are query suggestion, term substitution, query classification, document understanding or re-ranking of search results. User analysis tries to understand all aspects of a user. This may be understanding the information needs of individual users or whole user groups, clustering users or predicting their (navigational) behaviour.

2.5.1 Recommendation and Personalization

Recommendation is probably one of the most important applications of web usage mining in general, and of the implementation of user-behaviour data in particular. According to Quadrana et al. [212], recommender systems belong to the most successful applications utilizing user data to enhance systems in practice. There are several reasons for this: one reason is commercial, in that recommendations help users navigate to find popular items, which often goes hand in hand with with a personalization system that only recommends items the user is personally interested in [212].

The success of these systems is explained by the sheer amount of information users have to deal with today. As Batmaz et al. say, 'people confront a colossal amount of data sources which confuses them to find useful and appropriate content and results in the information overload problem' [16, p. 1]. The information overload may have many negative implications on user behaviour from a system point of view, which is why recommendation systems function as a way of filtering all of the available information. Users get recommendations for interesting items, which leads to a better user experience because they can directly jump to engaging content.

In theory, the content or items that are recommended could be anything. The range of applications is fairly wide. Items could be news articles, music, books or even travel tips. Depending on the system, any content can be recommended. In e-commerce systems such as the case study experimented with in this research, recommended items would be categories or products.

In general, there are three different ways to incorporate recommendation algorithms into a system. These are based on the way the algorithm makes recommendations, or more specifically, what kind of data the algorithm uses. Typically, the systems are split into multiple types [2, 16]: content-based, collaborative filtering and hybrid methods.

Content-based recommendations are the most direct approach to recommendation. The algorithm collects information about items and users and creates profiles that may be of interest to the reference user at hand. The goal is to recommend items that are similar to items the user has interacted with in the past. In general, interactions signalling a positive outcome are meant here, for example, buying products in the same category, watching multiple movies of the same genre or rating books from a certain author.

These examples make it obvious that the input for any algorithm is heavily dependent on the nature of the user's interactions. For example, in a recommendation for a movie, it does not necessarily have to be the same genre – the input could also consider runtime, actors, directors or even language or country of production. The list of possible attributes is almost inexhaustible and affects the quality of the input data and therefore the algorithm: whether a system uses unstructured or structured data, numeric or categorical variables is important [206].

Learning these attributes from the interaction histories on a per-user basis has several downsides. Content-based recommender systems are able to learn some kind of user profile and are effective at making new recommendations on the basis of this, but they usually cannot make new personalized user predictions [16]. They have difficulties in modelling specific user interests when there is not enough information to distinguish said interests [206]. A solution to circumvent this problem could be collaborative filtering. In classic collaborative filtering systems, users are recommended items that are preferred by other users [71]. The underlying assumption is that users who liked the same items at some point in time will probably enjoy similar items in the future [228]. Hybrid approaches combine the internal concepts of both systems to avoid their respective limitations [16].

Recent advances have focused on the implementation of deep learning algorithms on recommendation tasks. The introduction of algorithms that benefit from big data and the rise of new hardware such as more advanced graphical processing units lead to new possibilities not only regarding algorithms but also regarding the input data. Traditionally, academic research is often based on the typical user-item-matrix problem, considering only one type of interaction feedback per pair [212]. New types of input are now considered that were not used before.

Quadrana et al. [212] introduce sequence-aware recommender systems, that deal with sequences instead of matrixes. Sequences mean, for example, the interaction history of a user (usually in the form of a multidimensional vector instead of a matrix). They emphasize that sequential data is far more common in practice compared to the theoretical problems that are worked on in academic research. With this statement in mind, the authors try to categorize sequence-aware recommendation systems into the following classes: Last-N interaction recommendation, and session-aware and session-based recommendations.

Last-N interaction-based recommendation systems would use a specified number of interactions for future recommendations. They state that such limits may be necessary in a situation where certain past interactions are not relevant for future interactions [212]. Session-aware recommendation uses information about past sessions and also the current session of the user to make recommendations. A session-based recommendation would only use the current session as input to make predictions and recommendations for the user.

These three recommendation types are exemplary for this dissertation, since they deal with exactly the kind of data that is researched here. Session-based recommendations assume that only the current session is known. As Wu et al. [283] put it, most systems assume that the user's history is known, although that is often not the case in real production environments. But recommendations will most likely be improved when the user's actual state up to the current moment is known [235], which is done in the session-aware recommendations.

Session-aware and last-n recommendations are equally interesting because they work with different underlying assumptions that have a direct influence on the topic of this research. Having differently formed segments would be more likely to have an impact on the outcome of the algorithm, which is the main hypothesis of both session-aware and last-N recommendation systems. This is especially true where sequential data is the input and there is a dependency between the values of a vector.

There are many different examples of algorithms that use sequential session data to feed their recommendation algorithms. The following overview makes no claim for completeness, since this is not the main topic of this research. Rather, the overview is intended to provide a glimpse into the state of research on (sequential) recommendation systems and how these systems work among the differing session concepts.

Rethinking the impact of different session-segmentation concepts is especially important since there seems to have been little forethought invested in the input used for algorithms [62]. Zhao et al. [294] have noted the lack of research into how modelling assumptions affect the quality or accuracy of a recommendation system. Their study looks at multiple modelling assumptions on user-interaction data and the effect these have on user recommendations. Although the modelling is different from the actual segmentation, the implication is still true: not enough research is being put into the input data. Epure et al. [73] present a good example of this. They use Markov processes to recommend news articles based on different levels of user behaviour. Short-, medium- and long-term reading interests are considered. In their definition, short-term only considers popular articles on a per-article level (i.e. staying in the same category), while medium- and long-term are depending on the user interaction history. They find that different combinations of these interest levels lead to a higher response rate and performance of the recommendations. Different parameters are considered to determine the segments used for the interest levels, but all user sessions are simply calculated with a 60-minute inactivity timeout. A similar goal is aimed for by Song et al. [236], who try to diversify categories in a personalized-content recommendation system. They estimate the interest of a user at different levels and stages of a session, using a 20-minute inactivity timeout to determine these sessions.

Wang et al. [267] follow a slightly different approach to tackle various problems of recommendation systems, such as the cold-start problem. They build graphs from user sessions in order to construct item embeddings, comparable to the word embedding concept proposed by Mikolov et al. [186]. All sessions used to compute these graphs were segmented by a 60-minute inactivity timeout. Building the graphs on different versions of the input would most likely lead to different results.

Despite some other approaches, the overwhelming majority of recent articles dealing with recommendation tasks focuses on the utilization of deep learning methods. In particular, the implementation of recurrent neural networks (RNNs) seems to be the main direction of research because they are a perfect fit for sequential data. As Patterson and Gibson [205] explore, while RNNs belong to the family of feed-forward neural networks, they are technically able to include temporal dependency between steps or values. Being able to model temporal dependency in sequential data is ideally suited to this kind of task. It is superior to comparable techniques such as Markov models because RNNs can incorporate long-term dependencies in the input data much better.

The amount of literature using RNNs is consequently high. The first authors to apply RNNs to this domain were probably Hidasi et al. [102] in 2016. They proposed an RNN for session-based recommendations and tested it on two datasets with predefined session boundaries. Despite delivering good results, the model is built solely on the basis of predefined sessions. It is unclear how these sessions are constructed, which might suffice for session-based algorithms but makes it hard to measure the impact of session boundaries in other applications. The improvements brought by RNNs compared to other models used in these kinds of tasks are notable, nonetheless. Shortly after, Twardowski [252] proposes an RNN architecture that utilizes contextual item and event information from the dataset. Information about how the user interacts with items is fed into the RNN and used to make predictions. Sessions are defined as uninterrupted sequences of activity with a 30-minute inactivity timeout. The contextual information improves the results compared to baselines (logically depending on the richness of the dataset).

Ruocco et al. [225] also note the effectiveness of RNNs for session-based recommendations. They work on a problem comparable to the common cold-start problem – when only the current session is available as input it is hard to produce valuable recommendations at the start of a session. To overcome this, the authors present their idea of developing a second RNN that is capable of learning a representation of recent sessions to predict the interests and a starting point in the coming session. They call this the inter-intra RNN. Vector representations of previous sessions are used to model the output of the first RNN (the inter-session RNN), which they use as the initial state of the intra-session RNN. They experiment on two different datasets. One from Reddit[21], where they applied a timeout heuristic of 60 minutes to model sessions. The other from Last.FM[22], where a 30-minute timeout was applied. Afterwards, they did some fine-tuning to make their algorithm work. The first step was to remove consecutive repeated actions. Since RNNs need a maximum

[21] `https://www.reddit.com/`, retrieved 5 December 2021.

[22] `https://www.last.fm/`, retrieved 1 December 2021.

length of sequences, they also set the maximum length of sessions to consist of 20 interactions. Sessions consisting of over 20 and under 40 interactions were split into two sessions. Longer sessions were assumed to be bot sessions, and thus were removed from the dataset. Also, sessions with only one interaction and users with less than three sessions were removed. Compared to the other models, their architecture considerably improves the outcome, especially regarding the cold-start problem. Interestingly, despite using an RNN for the reason that it perceives the notion of sequential order and takes the importance of that into account, the authors do not put much effort into the session generation. This is even more important since they use the session representations in their inter-session RNN, therefore possibly changing the outcome completely. They note that 'many other models use the relaxed assumption that the order does not matter' [225, p. 2], but do not acknowledge an important parameter for this. Their work was partly reproduced in Section 6.2 as part of the evaluation process of this dissertation.

Very similar to this idea is the algorithm presented by Quadrana et al. [213] from 2017. They also use a hierarchical RNN where one layer is supposed to model an inter-session representation, personalizing the intra-session RNN with cross-session information. Sessions are again defined as groups of interactions within a certain time frame. Differences are to be found in the representation of the previous sessions, where Quadrana et al. only used the last hidden state of the single intra-session RNNs as well as using some different parameters when transferring this information to the current intra-session RNN. The model was tested on two datasets as well. One is from Xing[23] (from the RecSys Callenge 2016[24]), the other is a proprietary dataset from YouTube[25]. The logs were manually divided into sessions with a 30-minute inactivity timeout. Preprocessing was similar to Ruocco et al. [225], as the authors decided to delete repeated actions within a session and low-frequency interactions. Sessions with less than three interactions were removed as well as users with less than five sessions. Evaluation was done against several baselines with Recall@5, Precision@5 and MRR@5 (mean reciprocal rank). The authors reported strong improvements compared to the baselines. They argued that the length of user history has a significant effect on the quality of recommendations. They also analysed the session density, measuring the impact of events in a session on recommendation quality. Results indicate that, depending on the properties of the dataset, longer sessions tend to work better.

Again, very similar, Vassøy et al. [257] proposed a joint model based on a hierarchical RNN. They used the same approach as Ruocco et al. [225], using an RNN to create an inter-session representation and propagate the output to the intra-session RNN. Additionally, they added context representations for items, inter-session gap-time and the user, as well as a model for time and loss. A different approach is presented by Ren et al. [220]. Here, the authors argued that keeping a complete representation of long-term behaviour in mind when predicting future user interactions might lead to improved results.

[23] `https://www.xing.com/`, retrieved 15 November 2021.

[24] `http://2016.recsyschallenge.com/`, retrieved 15 November 2021.

[25] `https://www.youtube.com/`, retrieved 15 November 2021.

They provided a memory network model that maintains a hierarchical memory storage for every user, updating it periodically, while basically utilizing all sequential behaviour by a user. Similarly, Pi et al. [208] argued for the separation of long-term user interest modelling and prediction. They presented a 'user interest centre' capable of storing and updating interests on unlimited length sequences and a memory-model to use for real-time predictions. The same is true for Kaya and Bridge [126], who provided an intent-aware recommendation system in the domain of movies.

There are many other examples that employ recommendation algorithms on the basis of sessionized data. As has been shown, most articles use a simple session definition, often employing a temporal mechanical approach. Other articles test their algorithms on data with sessions where there's no clear information about how these sessions are defined: [6, 11, 15, 22, 26, 101, 105, 130, 134, 136, 150, 152, 174, 221, 235, 251, 253, 264, 285].

Overall, several problems can be identified here. For one, many of these algorithms rely on the sequential structure of the data, therefore implying a dependency between the individual values of the sequence. The majority use some form of mechanical session separated by a temporal inactivity constraint. This is a clear restriction on the input data, which is highly likely to have an unknown impact on the outcome. For strictly session-based algorithms, the limitation on temporal sessions may be suitable as these types of recommendation algorithms will get their input most likely on the fly while the user is browsing: having always only the current interaction sequence of a user as the input may make the segmentation of said input less important. Nevertheless, the segmentation still may have an impact on the training of any algorithm. Arbitrarily dividing sessions by an arbitrarily chosen timeout might influence the outcome in unforeseeable ways. Additionally, training an algorithm on data where the session definition is not known is questionable.

2.5.2 Making Queries Work

Comparable to recommendations, IR as a general area has seen a lot of research into the utilization of interaction logs. Considering the relationship between user, issued queries, retrieval system and documents, it is reasonable to use the interaction logs of the user to improve the queries and, ultimately, the retrieval system. The following section shows some insights into how queries can be improved using information from sessionized data.

The possible applications for sessions in the area of query improvements are wide. There are many easy targets for improvement using even the short-term history of a user. The same point of criticism as mentioned before is still valid: using the same session definition for every application is questionable and may not correctly reflect the quality of the algorithm.

The original idea of improving queries is an intuitive continuation of IIR. Having to hand multiple queries and potential reformulations leads to a variety of possible applications. Query expansion and substitution, query classification and the re-ranking of search results are discussed briefly below.

Query expansion, term substitution or query suggestions are probably the most common use cases. These all boil down to improving the user's current search terms to hopefully achieve a better retrieval performance. This could be either by automatically expanding the query terms via analysing implicit feedback on the search results [56, 57], using [107] or generating new [121] terms from similar older queries, re-ranking results by utilizing implicit feedback [3, 118, 119] or clustering queries by result co-occurrences for new query suggestions. The use cases here are implemented using different strategies, as depicted in the following examples.

Xiang et al. [286] presented a context-aware ranking system, for which they devised multiple principles describing the relationships between adjacent queries in a session. Sessions are segmented by a 30-minute inactivity timeout. Applied to ranking models, the results seemed promising. Shen et al. [230] propose a theoretic framework for utilizing implicit feedback to user and context models. To achieve this, they utilized the immediate search context, meaning the preceding queries and any clicked documents. They compared representations of the queries – consisting of title and snippets for the first 50 search results – using cosine similarity to circumvent vocabulary mismatch in favour of using query text alone. Their algorithm is dynamic, updating relevance ranking of unseen documents based on the collected implicit feedback of the user. They trained a client-side agent that creates a user model and acts as a support, updating its own statistics about the user model and the queries, and which re-ranks documents accordingly.

Huang et al. [107] use a five-minute timeout in their study. Based on a study of 2,369,282 query transactions from several popular search engines from Taiwan, the authors introduced a new recommendation algorithm based on the search context from similar user sessions instead of using the documents from the query results.

Filali et al. [77] focused on generating query reformulations using the search context of a user to get better search results for ambiguous queries. They modelled a query history-reformulation algorithm that updated the reformulation scores of a set of candidate reformulations to enable removal of less relevant queries from the list. The algorithm compared the similarity of rewrite candidates to the current query with the (average) similarity of said candidates to all related queries in the search history, adding weight based on the similarity of the old queries to the current one. It used the search context as implicit feedback for adding new candidates. Their model was based on an arbitrarily long search history – it can be as long or as short as needed, there are no restrictions. In their evaluation, they analysed one month of search logs from Yahoo with two million automatically labelled examples, alongside a sample of 600 history-reformulations assessed by experts. The search history was summarized with a most relevant bag-of-words-model. They found giving weight to the search history beneficial: In their experiments, they found that the relation of candidate queries to history was as important as the similarity to the current reference query.

Cao et al. [37] presented a context-aware query suggestion algorithm. First, their algorithm mined so-called query concepts to avoid sparsity of context-data. Therefore, a click-through bipartite graph was developed, where query nodes represented unique queries, URL nodes represented unique URLs and edges between them were created if URL u was clicked after the a query q. Weight was given by the number of clicks over the complete log. Afterwards, they ran a self-developed clustering algorithm to find similar queries and put them into a query concept. Then, sessions were segmented from their logs using the 30-minute inactivity timeout to build sequences of queries, which were then modelled into a concept sequence suffix tree. They experimented on a proprietary commercial search engine dataset with 1,812,563,301 queries in 840,356,624 sessions. The results indicated that their approach performed well compared to simple baselines (for example, adjacency in the overall log to the reference query, n-grams related to the reference query).

Many other models use either Markov processes or also RNNs. Often, these types of algorithms are applied to create context embeddings that can then be utilized for finding similar query terms or to improve the understanding of the search context. For example, Dehgani et al. [60] tackled the problem of query suggestion in a session-based context by using the user's current session to generate new query terms. They embedded the session context with sequence-to-sequence embedding using an RNN. Sessions were defined via 30-minute inactivity timeouts. Similarly, Mehrotra and Yilmaz [181] used task embeddings to learn query representations that were then used to generate better query suggestions. They used predefined sessions without clearly stating how these were defined. Kannadasan and Aslanyan [123] worked on personalized query auto-completion using within-session embeddings from queries. Tested on an eBay[26] dataset using the common 30-minute inactivity timeout, the authors embedded multiple features to model the user context, revealing significant improvements. Völske et al. [262] introduced a new abstraction of task detection to enhance query suggestions. In their study, they mapped reference queries to already identified logical sessions with task notations, defined across all user queries related to a task. To support individual users, all data that related to a task were leveraged. They tried several approaches for the mapping, including calculating the WMD based on word embeddings.

Halder et al. [90] proposed a sequence-to-sequence-based neural architecture to identify the information needs of users. Their model is based on the query history of the user within one search session as well as on representations of the results the user explored. Their reasoning was that a search session is like the conversation between two agents: the user with their information need reflected in the queries and the search engine with its respective result sets. Having received an answer, the user might want to edit the query, thereby creating what they call 'query edits'. These edits are defined by the removal or adjustment of query terms. The authors estimated the probability of the retention of each query term based on the user's past interactions during the current session, essentially by selecting the important terms in the current query. By being able to predict which words might be dropped, they declared that the remaining words were more aligned with the information need of the user. For their experiments, real-world data sets from AOL and Yandex were used. Although exploring vastly heterogeneous data regarding demography

[26] `https://www.ebay.com/`, retrieved 5 January 2022.

and temporal characteristics, their experiments relied solely on the given session definition of the datasets, thereby limiting their predictions to single, self-contained visits. Their preprocessing also added to that problem, as they kept only a specific subset of sessions. To test their proposed methods, Halder et. al. compared multiple baseline models with their own approach on two tasks: 1) predicting how a user would edit a query and 2) selection of the next query out of a given list of possibilities. Their approach outperformed the tested baselines.

The variability of algorithms is as wide as the area of recommendation. The same is true for the applicability of many results in this area, whereby, frequently, the authors implement their approaches in different systems. The outcome of these algorithms appears to be versatile and useful for different use cases. The flexibility of the algorithms is impressive; it is therefore puzzling to see that the majority of algorithms are tested with mechanically identified sessions with arbitrarily chosen inactivity timeouts. When developing an algorithm that utilizes context information in any way, the said context should receive the same degree of thought as the algorithm itself.

2.5.3 Understanding Users and their Needs

The same dissonance can be observed in the area of user analysis. Generally, the research literature aims to better understand the user's satisfaction and behaviour. In the context of Information Retrieval, this is mainly defined as search satisfaction, which generally means the fulfilment of a user's information need [78]. Studies include those that try to understand the level of satisfaction among users [78, 93, 94], aim to predict the actual interest or information need of the user [273] or analyse and understand user behaviour to improve the website [277]. Other studies analyse interaction logs to more accurately describe user behaviour [255] or cluster different user groups or even sessions [9, 248, 292]. Users' experience on different devices is the subject of research as well [92, 284]. Another important direction is the prediction of the next interaction in a sequence of interactions [23, 204, 234, 292].

The latter is of particular importance to many modern systems. Being able to predict the next interaction helps systems to improve their support of the user. This could be reflected in actively guiding the user to the content they are looking for or in knowing when to display certain advertisements. For example, if a system could predict that a user was considering buying a product in a session with a certain probability, it could compliment that behaviour by simplifying the process to perform a purchase.

Many studies have attempted to describe user behaviour. Following research into online shopping behaviour [24, 110], Moe et al. [187] described two basic types of search after analysing clickstreams: exploratory search and goal-directed search. While the latter refers to focused behaviour with a planned purchase in mind or in order to collect information to make a potential purchase, the former might be less specific and directed. The authors described this as 'stimulus-driven' instead of 'goal-driven', whereby any stimulus might lead to impulsive purchases. The authors combined these underlying behavioural types with the likelihood of making a purchase, resulting in four categories of shopping strategies: directed buying, search/deliberation, hedonic browsing and knowledge building [187, p. 30]. To enable classification of sessions into these categories, they devised a set of session-level metrics and performed a k-means cluster analysis of the data, coming up with four to five clusters. Unfortunately, it is not clear how they defined their sessions, which makes the results unreliable.

White and Drucker [275] provided analysis of user-behaviour variability based on log files, for which they extracted ‘search trails’. These are segments starting with a (directed) query and ending with a defined point of termination, for which they used some basic assumptions: for example, returning to the search engine homepage, checking email, visiting a bookmark and a 30-minute inactivity timeout. The experiments were performed by tracking the interactions of 2,000 users over a period of time, resulting in insights into different behaviours such as particularly consistent or particularly variable interaction behaviour. The same definitions were reused later by White and Morris [278] and White et al. [272, 276] to research the behaviour of search engine users and predict interest models.

Bandari et al. [14] proposed a method for categorizing user sessions. They considered every session as a document and every interaction as a term within this document. Their approach included finding the term frequency–inverse document frequency (tf-idf) weights for every interaction in a session, normalizing the resulting vector, reducing dimensions via principal component analysis (PCA) and clustering the result vectors with a k-medoids algorithm. Afterwards, the resulting clusters were classified to be able to interpret the outcome regarding specific activities on the platform. Sessions were segmented by a timeout calculated via the inter-activity time distribution of user events.

A common, more modern means of achieving reliable user representation or, more specifically, a session representation, is by again using embeddings. Context embeddings are, just like in the fields of recommendations and supporting query performance, the preliminary method used to determine user behaviour. The predominant methods used to create sequence-to-sequence embeddings seem again to be RNNs and LSTMs.

A recent example is presented by Bigon et al. [23], who employed an LSTM architecture with additional steps to predict whether a user is likely to make a purchase within a session. They also compare the output of the language models with Markov chains. The input data comes from a European e-commerce website, segmented into sessions with a 30-minute inactivity timeout. The research of Gu et al. [86] looks at hierarchical user profiling to identify interests in different granularities. As categories and products are often organized in a tree structure, a system should be able to provide different granularity of recommendations based on user behaviour. Therefore, they present a Pyramid RNN with a Behaviour-LSTM. Every experiment was based on predefined sessions, the boundaries of which are unclear.

Other articles utilize Markov processes instead of language models and RNNs. Patil and Patil [204] tested multiple Markov models to predict browsing behaviour. Sessions were segmented by the common 30-minute inactivity timeout. Based on the work of Jones and Klinkner [120], Hassan et al. [93] used the same hierarchy of search tasks and search goals, and the procedure of automatically segmenting interaction logs into this hierarchy, to predict the goal-level success of search tasks. Using a Markov model, they were able to evaluate individual user's search task success instead of having to analyse a generic user and generic relevance measures at query-level. Cherniak and Bridgewater [48] modelled variable-length Markov chains, augmented with the inclusion of browsing intents, to predict buying behaviour. For their experiments, they assumed that users pursue exactly one certain goal per session. Every session is constructed with the usual 30-minute inactivity timeout, using data from seven days of user logs from eBay[27]. They compared two approaches: each buyer session representing a single intent, and each individual buyer session as a model event to predict the next interaction in a user's path, raising accuracy by 300% when using the first approach. They inferred with these results that it was better to consider modelling sessions as a whole than to look at individual actions.

This literature review has indicated the wide variety of methods in use. Sessions are used as input data for all kinds of applications, whether to support user behaviour or to try to understand it. Much of the research focuses on the algorithms though, not so much on the input data. Many examples simply use the 30-minute inactivity timeout or a variant without considering potentially different results with a different sessionization.

What follows in this dissertation provides a comprehensive overview of the different session-identification approaches, comparing them using three different use cases, to prove that the input data has a significant impact on algorithmic performance.

[27] `https://www.ebay.com/`, retrieved 5 January 2022.

Chapter 3

Case Study: E-Commerce Platform

3.1 System Overview

This dissertation addresses the research questions with a case study. Using the example of an information system on the Internet, the proposed methods are tested and all results evaluated. The case study is based on data extracted from log files on `www.idealo.de` (retrieved 10 December 2021). This German price-comparison e-commerce platform, maintained by idealo Internet GmbH (company with limited liability), was founded in 2000 with the self-proclaimed mission to support users in finding the best price when buying online products. Since then, the company has grown, employing over 1,100 employees and with an Internet presence in multiple countries. The German website is among the most popular e-commerce portals in Germany, attracting around 1.9 million visits per day. The overall strategy is to be the starting place for online shoppers – users should always start at idealo and check for prices during their journey. The company claims to be a neutral and transparent provider of information about the market situation of any product; as they do not allow shops to buy higher positions in their price lists, only the actual price comparison of products is relevant to shoppers before placing their order at the shop with the best offer[1].

The website is a pure price comparison platform. The company does not run their own warehouses or sell products as as an online shop would. As a price comparison platform, they only list offers from online shops. Users can visit the system to find the best price for any given product and use idealo as a stepping stone to the shop with the best offer. From a content point of view, the website is basically a universal catalogue. The database contains entries for probably every type of category and product purchasable on the Internet. These entries are maintained by subject-specific content management teams in order to offer not only the best prices but also comprehensive information about the product itself. The inventory and the related content are, therefore, one of the most important aspects of

[1] Information is taken from `https://www.idealo.de/unternehmen/ueber-uns/`, retrieved 1 November 2021.

the business. Only if the website provides an adequate and up-to-date overview of the market will it be able to assist in any potential purchase a user might want to make. The investment in keeping up with new trends and offering not only pricing information but also product information, provides many possibilities for interaction during any shopping journey.

The company strives to inform their users about all aspects of a product or products before they buy anything. As one of the most popular e-commerce websites in Germany, the platform receives huge amounts of traffic. This has led to an addition to the business model, away from a pure price comparison platform. Originally, the idealo business model consisted only of shops listing their offers, meaning that every time a user clicked on an offer the shop payed a small commission to idealo. However, since 2015, idealo has added the new *Direktkauf* (direct purchase) feature. This new model is comparable to an online marketplace where users can make orders via idealo as the facilitator. This means that idealo then takes over and runs the complete transaction with the shop. The main benefit for the user is that only one contact – with idealo – is needed, not multiple accounts with multiple shops. Since its introduction, the *Direktkauf* has seen steady growth and is now another important business case in combination with the classic transfer-to-shop model.

Aside from the core concept of the price comparison, idealo has some other notable features. The most prominent one is the idealo account. It allows the registered user to set up a wish list and to save all their information for orders in one place. All related information is also synchronized between different devices – mobile phones, for example. The wish list provides an overview of all saved products along with a price history and the current price. There is also the possibility to adjust another of idealo's features – the price alert. The price alert is one of the oldest services provided by the portal. This feature allows the registered user to be notified via email if a product reaches a specified price. It adds another layer to the self-declared goal of offering the user assistance in any phase of the buying decision. Aside from the website, idealo also maintains applications for iOS and Android. In appearance and usability, those are very close to the web presence.

3.2 Website Structure

The following section describes how the website is structured, explaining every significant page type and its function. As the portal is constantly in development, the pages are living objects, which get updated from time to time as required by the business. This might lead to differences in description and the actual appearance online. This also applies to the different pages themselves – the portal might create new page types or delete others. The following description therefore applies only to the dataset used in this dissertation, which contains the complete year 2018.

In theory, the current state of idealo resembles the basic concept of a department store. Where one would normally have thematically related groceries in a structured physical environment, idealo provides a similar set-up online. There is no physical representation

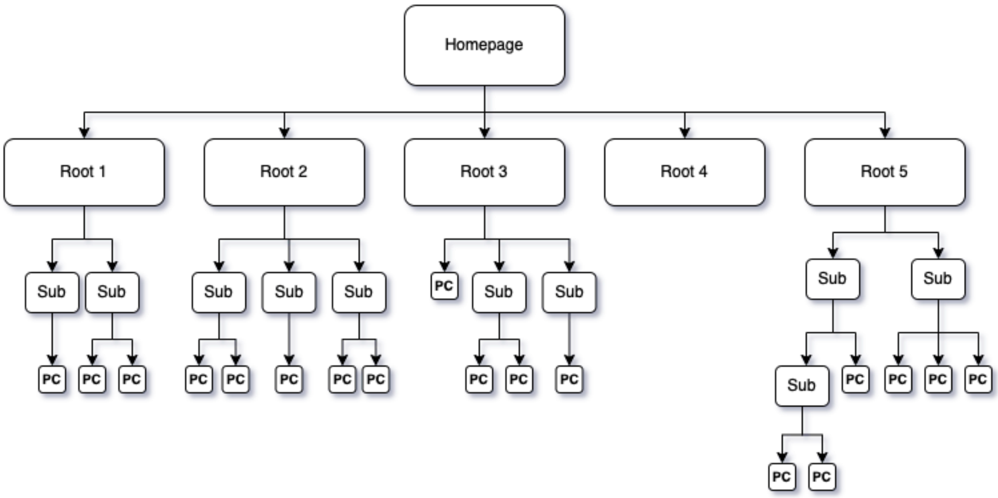


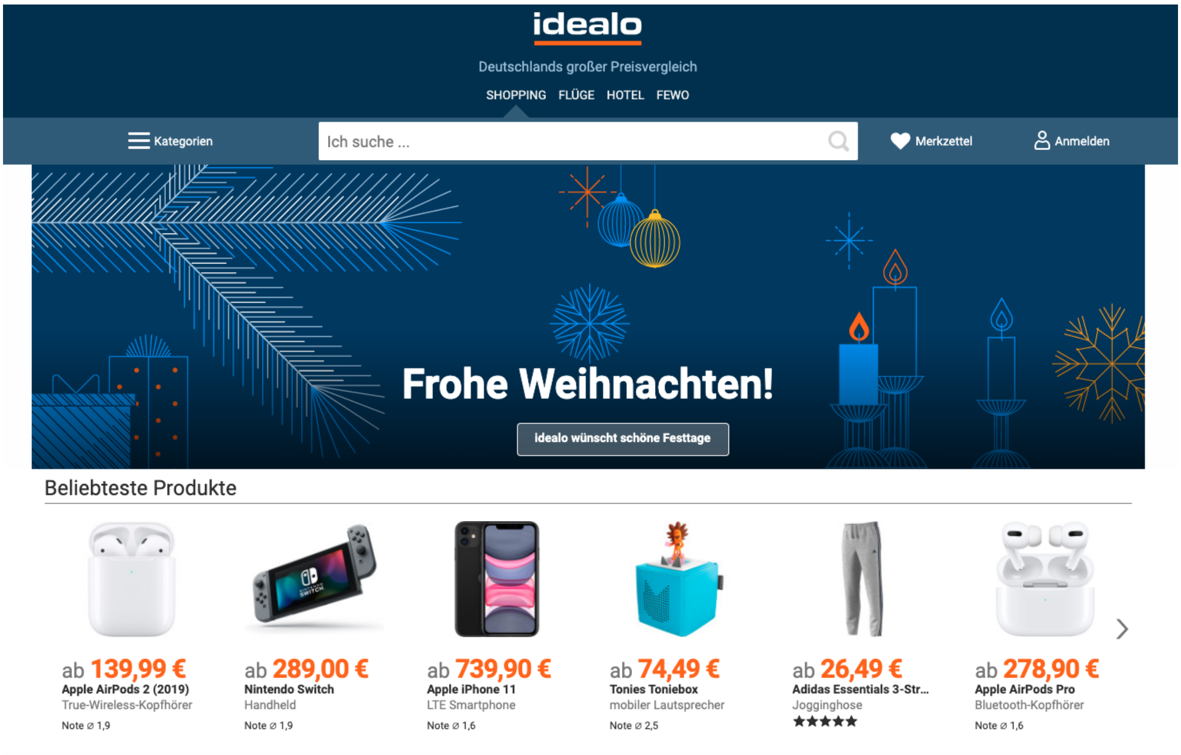
Figure 3.1: idealo Category Tree: Root = Root Category, Sub = Subcategory, PC = Product Category.

as such, but the website structure can be visualized as a large tree with multiple branches. Figure 3.1 depicts this structure.

The index page is the homepage. All other pages are connected to this page. The homepage allows access to all other pages as well as to organizational and administrative pages like the imprint, job offers or the sitemap. Below the homepage are the different levels of the tree. The first level consists of the root categories, marked as 'Root' in the figure. There are multiple root categories with different topical and technical specifics. Below the root level are usually the subcategories, marked as 'Sub'. These represent another level of detail and divide the root categories into more precise topical areas. The lowest level consists of the product categories, which contain actual product pages, marked as PC. The different levels and eventual special cases will be described in the following paragraphs.

The homepage is the anchor page for all other levels. As visible in Figure 3.2, the prominent idealo header dominates the site. Below the logo are quick links to the other idealo domains: flight booking and travel accommodation. It provides the user with multiple possibilities for navigating through the portal. On the left is a drop-down menu opening to navigation of the different root categories linked to the homepage. Every root category has its own row with quick links to the most popular categories. This enables the user to easily browse along the category tree – either by starting on one of the root categories or by directly diving deeper. In the middle is the search bar. To the right of the header are links to the wish list and the user account page.

Below the navigation and the header is a seasonal teaser, advertising specific categories that are relevant at the time. In Figure 3.2, there is a banner for Christmas. Most of the time, these teaser banners provide a possibility to navigate quickly to popular seasonal categories. The rest of the start page consists of different objects supposed to help the user in browsing the page and be able to quickly reach the desired pages. In Figure 3.2, there is an object for the most popular products. When revisiting idealo on the same



ab 139,99 €
Apple AirPods 2 (2019)
True-Wireless-Kopfhörer
Note ⌀ 1,9

ab 289,00 €
Nintendo Switch
Handheld
Note ⌀ 1,9

ab 739,90 €
Apple iPhone 11
LTE Smartphone
Note ⌀ 1,6

ab 74,49 €
Tonies Toniebox
mobiler Lautsprecher
Note ⌀ 2,5

ab 26,49 €
Adidas Essentials 3-Str...
Jogginghose
★★★★★

ab 278,90 €
Apple AirPods Pro
Bluetooth-Kopfhörer
Note ⌀ 1,6

Figure 3.2: Example of idealo's homepage.

device or browser while not deleting saved cookies, these objects are supplemented with personalized recommendations related to the last visited products and categories.

From the homepage and according to what is shown in Figure 3.1, the category tree unfolds into different sublevels with the root categories being the first level. As the name implies, these root categories are the foundation of the tree, representing general topics of content. There are multiple root categories (13 different ones in the analysed dataset). They fulfil the function of a hub, containing several subcategories related to the respective content topic. Subcategories are the next conceptual level in the category tree. Usually, multiple subcategories are connected to one root category. Like root categories, subcategories can also contain subcategories as shown for Root 5 in Figure 3.1; there is no hard restriction on the amount of levels in the tree. The lowest level are always product categories though.

A special case is depicted in the fourth branch of root categories in Figure 3.1. This branch represents the so-called open catalogue. If an offer is too ambiguous or unique to assign it to a product or a category, it will be mapped to this root category. There are no subcategories, product categories or products in here, only unassigned offers. This root category is only accessible via search – there is no way of browsing to it.

Figure 3.3 shows the root category *Elektroartikel* (electronics) with the related subcategories. Below the idealo header at the top of the page is a navigational link chain, showing the level of depth according to the category tree. This offers another way of browsing the site, displayed on every other page as well (subcategories, product categories or products). On a root category page are several related subcategories like *Computer* (computing) or

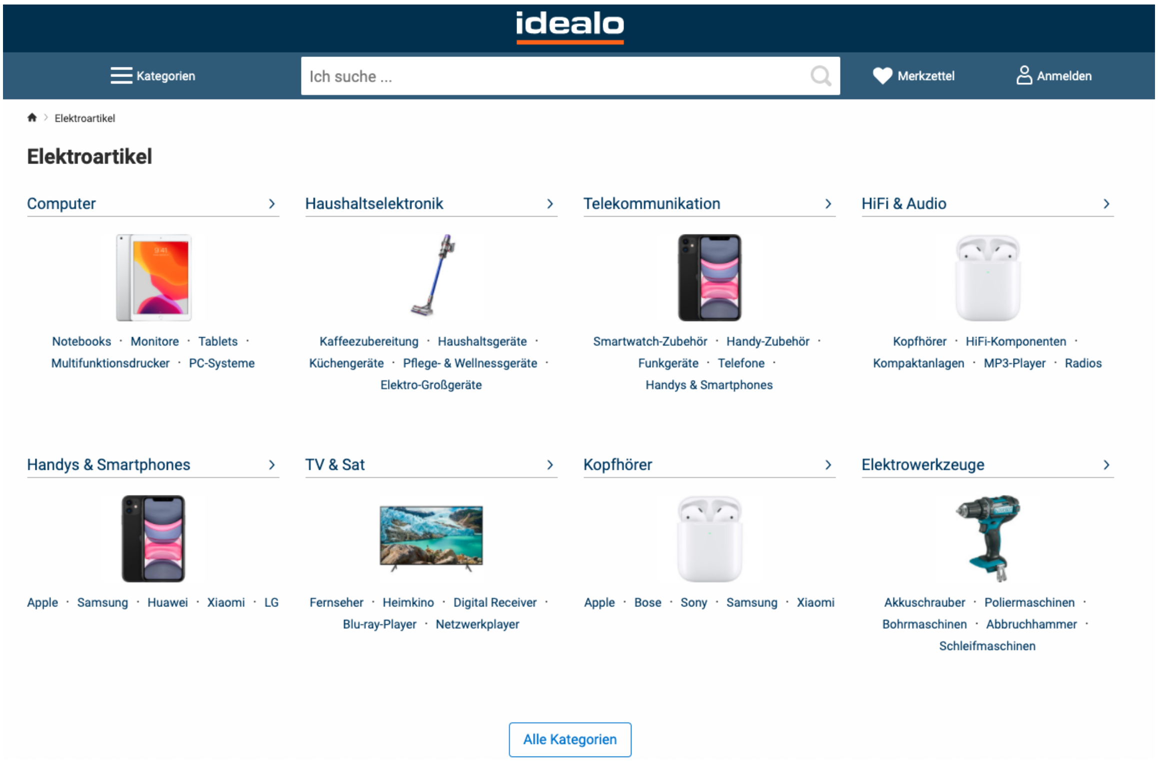
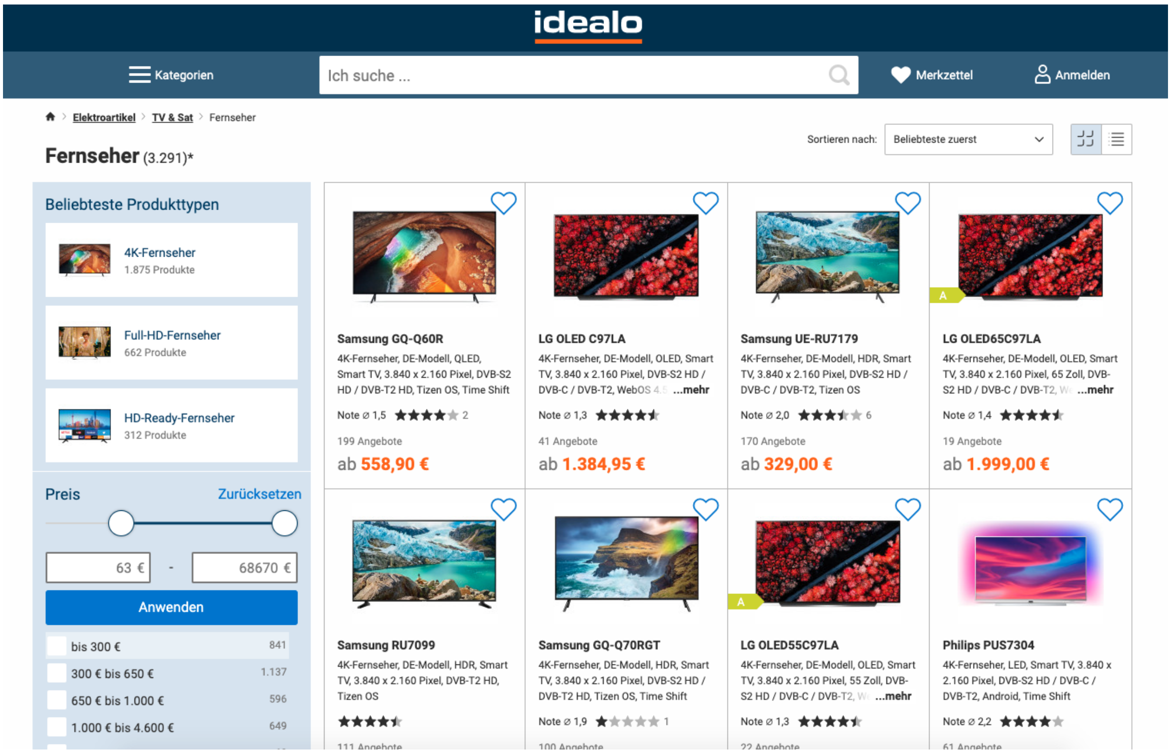

idealo
Kategorien
Ich suche ...
Merkzettel
Anmelden
Elektroartikel
Elektroartikel
Computer
Notebooks · Monitore · Tablets · Multifunktionsdrucker · PC-Systeme
Haushaltselektronik
Kaffeezubereitung · Haushaltsgeräte · Küchengeräte · Pflege- & Wellnessgeräte · Elektro-Großgeräte
Telekommunikation
Smartwatch-Zubehör · Handy-Zubehör · Funkgeräte · Telefone · Handys & Smartphones
HiFi & Audio
Kopfhörer · HiFi-Komponenten · Kompaktanlagen · MP3-Player · Radios
Handys & Smartphones
Apple · Samsung · Huawei · Xiaomi · LG
TV & Sat
Fernseher · Heimkino · Digital Receiver · Blu-ray-Player · Netzwerkplayer
Kopfhörer
Apple · Bose · Sony · Samsung · Xiaomi
Elektrowerkzeuge
Akkuschrauber · Poliermaschinen · Bohrmaschinen · Abbruchhammer · Schleifmaschinen
Alle Kategorien

Figure 3.3: Example of root category Elektroartikel (electronics).

Telekommunikation (telecommunications) as well as the occasional product category like *Kopfhörer* (headphones). From a design perspective, there is no real difference between the product tiles (images) or subcategories: for the first, top manufacturers are displayed below the category name, for the latter, the underlying categories are displayed. This can also be seen in Figure 3.1 in the example of the root 3. The root category here has different subcategories attached to it as well as a product category.

There are also different levels of subcategories. One of the iconic examples for this is the subcategory *Hifi & Audio* (home audio & HiFi), which itself is as big as a root category. Containing more subcategories like *Musikinstrumente* (musical instruments), this subcategory level is fairly widespread with a lot of additional downward levels. As can be seen in this example, subcategories are basically like root categories. There are no products, only additional subcategories or product categories at the lowest level of the category tree.

The structure of the idealo category tree is the result of a qualitative survey, research with focus groups and card sorting tests and also the work of the content department. The different category levels are supposed to feel as intuitive as possible. To achieve this, some categories are functionally connected to their original root but also displayed in a different branch to reach a broader audience. An example would be the product category *Babyschalen & Kindersitze* (car seats), which is originally connected to *Baby & Kind* (baby & child) but is also displayed in *Auto & Motorrad* (car & motorcycle). This is also the reason for some of the product categories at the root level.



Fernseher (3.291)*

Beliebteste Produkttypen

4K-Fernseher
1.875 Produkte

Full-HD-Fernseher
662 Produkte

HD-Ready-Fernseher
312 Produkte

Preis

63 € - 68670 €

□ bis 300 €	841
□ 300 € bis 650 €	1.137
□ 650 € bis 1.000 €	596
□ 1.000 € bis 4.600 €	649

Samsung GQ-Q60R
4K-Fernseher, DE-Modell, QLED, Smart TV, 3.840 x 2.160 Pixel, DVB-S2 HD / DVB-T2 HD, Tizen OS, Time Shift
Note ⌀ 1,5 ★★★★☆ 2
199 Angebote
ab 558,90 €

LG OLED C97LA
4K-Fernseher, DE-Modell, OLED, Smart TV, 3.840 x 2.160 Pixel, DVB-S2 HD / DVB-C / DVB-T2, WebOS 4.5 ...mehr
Note ⌀ 1,3 ★★★★★
41 Angebote
ab 1.384,95 €

Samsung UE-RU7179
4K-Fernseher, DE-Modell, HDR, Smart TV, 3.840 x 2.160 Pixel, DVB-S2 HD / DVB-C / DVB-T2, Tizen OS
Note ⌀ 2,0 ★★★★☆ 6
170 Angebote
ab 329,00 €

LG OLED65C97LA
4K-Fernseher, DE-Modell, OLED, Smart TV, 3.840 x 2.160 Pixel, 65 Zoll, DVB-S2 HD / DVB-C / DVB-T2, W ...mehr
Note ⌀ 1,4 ★★★★★
19 Angebote
ab 1.999,00 €

Samsung RU7099
4K-Fernseher, DE-Modell, HDR, Smart TV, 3.840 x 2.160 Pixel, DVB-T2 HD, Tizen OS
★★★★★
111 Angebote

Samsung GQ-Q70RGT
4K-Fernseher, DE-Modell, HDR, Smart TV, 3.840 x 2.160 Pixel, DVB-S2 HD / DVB-T2 HD, Tizen OS, Time Shift
Note ⌀ 1,9 ★☆☆☆☆ 1
100 Angebote

LG OLED55C97LA
4K-Fernseher, DE-Modell, OLED, Smart TV, 3.840 x 2.160 Pixel, 55 Zoll, DVB-S2 HD / DVB-C / DVB-T2, W ...mehr
Note ⌀ 1,3 ★★★★★
22 Angebote

Philips PUS7304
4K-Fernseher, LED, Smart TV, 3.840 x 2.160 Pixel, DVB-S2 HD / DVB-C / DVB-T2, Android, Time Shift
Note ⌀ 2,2 ★★★★☆
61 Angebote

Figure 3.4: Example of product category Fernseher (TVs).

In theory, product categories are the lowest level of the category tree. This type of category is basically comparable to the display window of a physical shop or the grocery shelves and cold cabinets in supermarkets. Figure 3.4 shows the general structure of the product category *Fernseher* (TVs).

Below the header, navigation and sorting drop-down menu are two objects – the product display and the filters. Being the lowest level, product categories contain products, shown as the tiles. The product tiles contain the same information regardless of category. There is the heart symbol for adding an item to the wish list, an image of the product, some product details always beginning with the product type, a rating if present, the number of offers and the price range. The product types are another level below the product categories – essentially splitting the products into thematically related groups. For the example category *Fernseher* (TVs), product types include *4K-Fernseher* (4K TV) and *Full-HD-Fernseher* (full-HD TV) among others. They do not have to be mutually exclusive, but may be.

On the left-hand side are filters. These are generated from the product information maintained by the content management teams. The content experts are responsible for choosing important filters that are relevant to users searching for certain products. They have to be meaningful from a content perspective as well as specific enough for the products and categories to be indexed by search engines. This is especially important for the product types so that users looking for *4K-Fernseher* (4K TV) on any search engine can find their way to the prefiltered *Fernseher* (TVs) site on idealo. Aside from search engine optimization (SEO), filters are a way to browse categories. By utilizing the product information, they essentially divide products into groups with different properties. A user can drill down the assigned products in a category to a set with attributes fitting to their needs.

Below the filters and the products on the category page, there is usually an advisory text or a buyer's guide. The intention of these texts is to introduce the content of the category, helping the user to find the right item. Depending on the category, they vary in length and depth. They may explain technical details, unclear differences between filters or recommend specific brands or products. Often, there is also a glossary at the end, explaining the most important category-specific attributes in short paragraphs for quick access. These texts do not only appear on category pages, but may also be displayed on pages filtered with any filter such as product types[2], manufacturer[3], or collection[4]. Whether a page receives a text is based on the popularity of the respective items or keywords. A keyword refers to a term with a high search volume on search engines. This naturally implies that the texts are not only written to guide the user but also for SEO purposes, as editorial content has a positive effect on the ranking of the search engine's results page.

The product categories contain the product pages, organized by product types in what resembles even more detailed, smaller categories. The product pages are the heart of the platform as they represent the core business of idealo. They are the most visited pages and the strongest ranking ones on search engines. Figure 3.5 on the next page shows a generic product page for an electric guitar. The figure shows the most common type of product page with no variants and no parent product. These 'non-varied' products have no direct relation to any other product. 'Main' products on the other hand are parent products containing various other elements – the 'variant' products, which are basically the same product in different styles or with slightly different attributes. They can differ from each other in colour, size, or any other descriptive attribute.

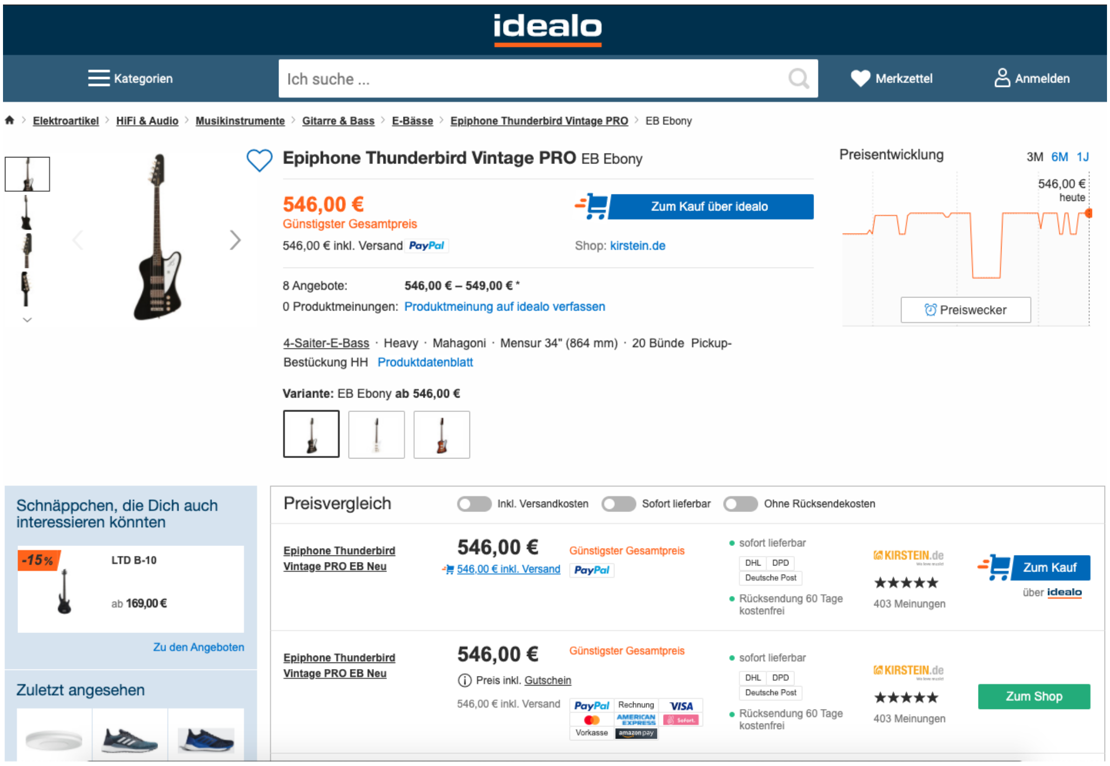
There are several objects on the page below the header and the navigation. Starting on the left is a big image linking to an image gallery, if present. Prominent is also the heart icon for adding the product to the wish list. On the right-hand side is the model name, put together by manufacturer and product title. Below the model name is a summary, containing the number of offers on the page and the price range, the number of user testimonials and the test review ratings. These objects are only displayed when present for the product. Below this, are the most important attributes and a call to action button for the complete overview of all product details.

On the right-hand side is the price development graph with the option to set up a price alert via the alarm clock icon. This interactive diagram shows the development of the lowest prices over three selectable time periods: three months, six months and one

[2] `https://www.idealo.de/preisvergleich/ProductCategory/4012F1921183.html`, retrieved 15 November 2021

[3] `https://www.idealo.de/preisvergleich/ProductCategory/3933F1451777.html`, retrieved 15 November 2021

[4] `https://www.idealo.de/preisvergleich/ProductCategory/5666F1499900.html`, retrieved 15 November 2021.



Elektroartikel > HiFi & Audio > Musikinstrumente > Gitarre & Bass > E-Bässe > Epiphone Thunderbird Vintage PRO > EB Ebony

Epiphone Thunderbird Vintage PRO EB Ebony

546,00 €
Günstigster Gesamtpreis
546,00 € inkl. Versand PayPal

Shop: kirstein.de

8 Angebote: 546,00 € – 549,00 € *
0 Produktmeinungen: Produktmeinung auf idealo verfassen

4-Saiter-E-Bass · Heavy · Mahagoni · Mensur 34" (864 mm) · 20 Bünde Pickup-Bestückung HH Produktdatenblatt

Variante: EB Ebony **ab 546,00 €**

Preisentwicklung

Schnäppchen, die Dich auch interessieren könnten

-15% LTD B-10
ab 169,00 €

Zuletzt angesehen

Preisvergleich ○ Inkl. Versandkosten ○ Sofort lieferbar ○ Ohne Rücksendekosten

Epiphone Thunderbird Vintage PRO EB Neu
546,00 € Günstigster Gesamtpreis
546,00 € inkl. Versand PayPal
sofort lieferbar
Rücksendung 60 Tage kostenfrei
403 Meinungen

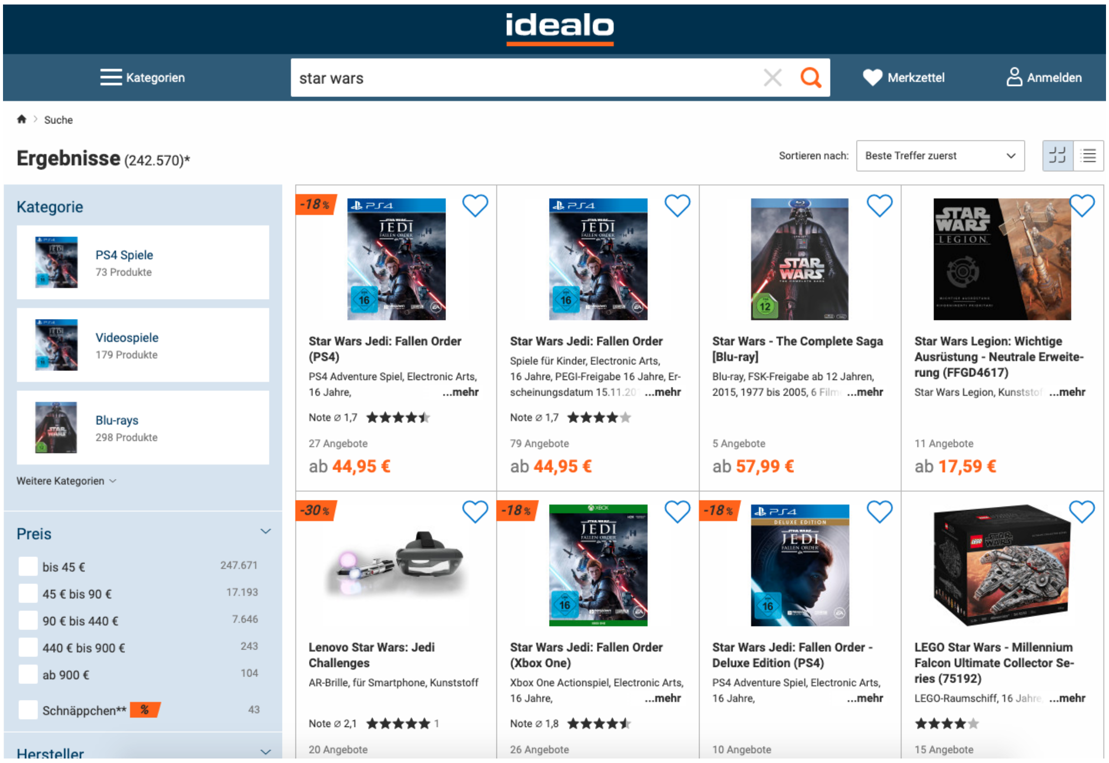
Epiphone Thunderbird Vintage PRO EB Neu
546,00 € Günstigster Gesamtpreis
Preis inkl. Gutschein
546,00 € inkl. Versand
sofort lieferbar
Rücksendung 60 Tage kostenfrei
403 Meinungen

Figure 3.5: Example of a product page for product Epiphone Thunderbird.

year. This makes it easy for users to see whether the product has a stable price history or, rather, a dynamic one, allowing them to estimate the optimum time to make a purchase. The price alert icon activates a pop up where users can input their preferred price to be notified via email when it is reached.

Below the objects on the upper left-hand side are elements to help users find other items of interest. These blocks encourage serendipitous findings as they show items similar, relevant or related to the product page and the browsing history of the user. The first object shows exactly this: the last seen products of the user. This is followed by the top 10 products for the primary type of that product by popularity. There is also an element showing the most popular products for the subcategory. The last standard object shows comparative prices from the international idealo sites. Depending on the type of product, there might be a block presenting the other variants.

To the left of these recommendation elements is one of the core elements of the product page: the price comparison list with all offers assigned to the respective product. At the top of the list are again some filter sliders for easier access to relevant offers: including the cost of delivery, ready for dispatch only, or the price without return charges. Below is the list of shop offers, which is actually at the heart of the page and the portal itself. Every offer has a separate rectangular tile. From left to right are the following elements in this tile: the title as given by the shop, the price with the additional information whether delivery costs are included, the payment options including a highlighted icon for the lowest total price, delivery speed and options, the shop logo with a rating which is also a link to the idealo shop page and lastly a call to action button. There are two types of these buttons. The



Ergebnisse (242.570)*

Kategorie

PS4 Spiele
73 Produkte

Videospiele
179 Produkte

Blu-rays
298 Produkte

Preis

□ bis 45 €	247.671
□ 45 € bis 90 €	17.193
□ 90 € bis 440 €	7.646
□ 440 € bis 900 €	243
□ ab 900 €	104
□ Schnäppchen** %	43

-18 %
Star Wars Jedi: Fallen Order (PS4)
PS4 Adventure Spiel, Electronic Arts, 16 Jahre, ...mehr
Note ⌀ 1,7 ★★★★★
27 Angebote
ab 44,95 €

Star Wars Jedi: Fallen Order
Spiele für Kinder, Electronic Arts, 16 Jahre, PEGI-Freigabe 16 Jahre, Erscheinungsdatum 15.11.20 ...mehr
Note ⌀ 1,7 ★★★★☆
79 Angebote
ab 44,95 €

Star Wars - The Complete Saga [Blu-ray]
Blu-ray, FSK-Freigabe ab 12 Jahren, 2015, 1977 bis 2005, 6 Film ...mehr
5 Angebote
ab 57,99 €

Star Wars Legion: Wichtige Ausrüstung - Neutrale Erweiterung (FFGD4617)
Star Wars Legion, Kunststo ...mehr
11 Angebote
ab 17,59 €

-30 %
Lenovo Star Wars: Jedi Challenges
AR-Brille, für Smartphone, Kunststoff
Note ⌀ 2,1 ★★★★★ 1
20 Angebote

-18 %
Star Wars Jedi: Fallen Order (Xbox One)
Xbox One Actionspiel, Electronic Arts, 16 Jahre, ...mehr
Note ⌀ 1,8 ★★★★★
26 Angebote

-18 %
Star Wars Jedi: Fallen Order - Deluxe Edition (PS4)
PS4 Adventure Spiel, Electronic Arts, 16 Jahre, ...mehr
10 Angebote

LEGO Star Wars - Millennium Falcon Ultimate Collector Series (75192)
LEGO-Raumschiff, 16 Jahre ...mehr
★★★★☆
15 Angebote

Figure 3.6: Example of a search result page with query 'star wars'.

green one is a link directly to the shop page. The blue one leads to the checkout funnel for the idealo *Direktkauf* – the user buys the item via idealo from the shop.

The category pages and the product pages represent the bulk of the content on the platform. They can be accessed via browsing starting from the homepage down the category tree, but also via the internal search. The search bar is prominently included in the middle of the header and is therefore easily accessible on any page. Technically, it is based on Apache Lucene[5] with some additional self-developed features. When issuing a search, the user can either directly hit enter for a free search or click on one of the suggested items. Additional features include a spellchecker, which automatically corrects minor spelling mistakes and a system that redirects unambiguous queries to fitting categories. For example, if a user issues the query *fernseher* (tvs), they are automatically redirected to the category *Fernseher* (TVs).

By using the search without the suggests, users can retrieve the third branch of content pages next to category and product pages. Every time a query is issued, a new dynamic result page is generated, containing a variety of products, offers and clustered offers from a range of relevant categories. Figure 3.6 shows a result list for the query 'star wars'. This example shows the mixture of results well. There are products from multiple categories displayed on one page with the look of a product category page.

Further down the list may not only be products, but also unassigned offers relevant to the search terms. In this case, 'unassigned' means that they are not connected to any product, but may be already assigned to a category. These offers are only accessible via

[5] `https://lucene.apache.org/`, retrieved 15 November 2021.

the search. In the result list, they are usually ranked below products for performance reasons. Clicking on these offers will lead directly to the shop. If multiple offers can be clustered together via article number or name, they are displayed as a 'cluster' page. These cluster pages have a similar appearance to product pages, but are dynamically created to produce a better overview of the results. Every tile in the result list counts as one result. For the 'star wars' query in Figure 3.6, there are 237,380 results, consisting of products, clusters and offers. Clusters and offers are not visible in the figure, but they have the same appearance as the product tiles.

When on the search result page, there are multiple ways to navigate further. By using the filters on the left-hand side, it is possible to drill down to specific categories or manufacturers or even price ranges. Down below, it is possible to navigate through the result list by clicking onto the next page. Aside from this, users can also issue a new query to the system.

Another form of search result page comes in the form of the so-called list pages. These are a special form of search result page, since they are dynamically created by the system to add additional reachable content for search engines. List pages are created by collecting important keywords and links from the manufacturers, product types, products and product categories to generate additional content. They are usually linked in a grey box at the bottom of product pages. As these are also indexed, they are therefore available to incoming traffic from search engines.

Aside from the actual content pages, there are also informational pages that can be visited by users. These include administrative pages, such as the privacy policy and informational pages about shops or manufacturers. On pages like the shop or the manufacturer overview, users can either search or browse shops or visit the manufacturers' info pages for further information about them. These pages may include contact information, short descriptions and reviews. These pages are vastly underrepresented in the dataset in comparison to the content pages and this is because they are not visited as often. It makes sense, therefore, that these pages are not indexed on search engines, since they are not directly related to the actual purpose of the website.

3.3 The Tracking Concept and Business Model

When a user visits idealo, nearly every page visited gets tracked via a self-developed Java tracker[6]. The backend server-side tracking results in JavaScript Object Notation (JSON) files sent through collecting servers into a data warehouse environment. As of 2018, Splunk[7] was used as a data warehouse. All JSON files are collected and indexed by timestamp for further processing. As this is backend tracking, only actual requests to the server are tracked. This means, only actual page views are tracked – every time a user clicks on a link or opens up a new page via search or any other possibility, the server receives a hit and the tracker records a new JSON file for the opened page. Not every interaction

[6] As of 2018, a self-developed tracker has been in use.

[7] `https://www.splunk.com`, retrieved 15 November 2021.

directly with a page is recorded, since in most cases there is no backend server involved. The following actions are among those that are explicitly excluded from the tracking as of the state of the analysed dataset:

- Clicking on an image or browsing through the gallery
- Any form of lists' sorting or filtering on the product page
- Interacting with the price development graph
- Receiving search suggestions
- Any scrolling on any page

Summarized, only actual page changes or page reloads are registered. Basically, the tracker differentiates between two types of content: internal page changes and lead-outs to external pages. The first type, known as page impressions, effectively relate to the viewing of a page. When a user enters any page on the website, a page impression is generated. This type of trace contains details about what users have seen on this page, where they come from and where they are right now. As the name implies, the trace is an impression of what the user has seen on the respective page. Lead-outs describe every click that leads the user from idealo to any other external page – the classic example would be the click on a shop offer. When a user initiates a *Direktkauf* (direct purchase) by clicking on the blue button on the product page, the trace that is generated is also tracked as a lead-out, although technically the user is not leaving the idealo environment.

This type of backend tracking only logs the navigational behaviour of a user along with the content on the pages visited. To complement the behaviour on the page, the logs generated by the tracker are enriched with so-called user actions. These are separately produced traces by production services to (partially) track the interactions with certain services. The most prominent example are the price alerts; when a user interacts with the price alert feature, a trace is generated.

Whenever a user visits the page for the first time or via an incognito mode of the browser, a new cookie is set. This cookie is not a user identifier, because it is set for every browser or device separately. It is not possible to connect multiple cookie values through different devices or browsers without additional information. A connection can only be achieved after a user has signed up for an idealo account and browses the site while logged in, and the same procedure must be followed on every one of the user's devices to install the cookies. While logged in, a hashed code representing the email address is added to the traces. Aside from this, every trace has a timestamp in milliseconds. The basic information a trace always contains are the following (technical) fields; they are necessary to create a data model according to the business model:

- user based (cookie value and HTTP user agent)
- system based (timestamp of the trace and media / domain information)

- content based (URL, HTTP referer and UTM parameters[8])

Cookie and trace time are used as the unique identifiers for every trace. With these two fields, simple session models are already possible, for example, with time constraints only. The other fields are either used for enrichment or are necessary to create more fields in additional steps. The HTTP referer, for instance, is necessary to identify the origin of a trace, while the URL is used for identifying the landing page, representing the first trace after the the user has visited the first page. The Urchin Tracking Module (UTM) parameters are important for marketing purposes to provide insight into where the user has navigated from. Often, UTM parameters like UTM medium, UTM campaign and UTM source are added to the URL; for example, a user arriving at the website from an email link may require these parameters to make clear that they have visited the website through this marketing channel. Together, the domain information and the user agent make it possible to determine the device and browser used to browse idealo.

Every trace generated by the tracker is stored in a data store. The files contain all user information in structured but unconnected traces. No information is attached that relates to which session the trace belongs. The files contain information purely relating to the user's activity, there is no actual preprocessing performed by the tracker. The foundation of the data model uses traces following these conditions, consisting of page impressions, lead-outs and user actions. The idea is to replicate only what the user has seen and intentionally done. Bots are excluded via a blacklist (using IP addresses and HTTP user agents). Further fields are built on this basis afterwards, mostly relating to attribution and content. These fields are specified by the second set of business rules, mostly referring to the marketing channels that are used for the correct distribution of the financial shares. As these attributional business rules go beyond the level needed as part of the evaluation of user behaviour which is the focus of this dissertation, they will not be considered. In this study's comparison of various modelling approaches, therefore, only the foundational business rules for the proper selection of user traces will be applicable. Furthermore, the data model used by the company for evaluating marketing campaigns will not be considered either, but, instead, everything in the following chapters will be modelled from scratch.

All in all, the data model is created specifically in order to be able to evaluate the business cases. Regarding the content of the web portal, the set of actual business cases is clear and simple. Basically, every conversion a user makes is considered a business case. The most important ones are logically the lead-out and *Direktkauf* models as these directly generate income for the company. Every time a user clicks out to a shop or orders something via *Direktkauf*, idealo receives a share via different compensation models. The user must initiate the checkout funnel to be able to place an order. For this, idealo does not receive any money, but as it still is a sort of conversion potentially leading to an actual

[8] Urchin Tracking Modules (UTM), a tag system for tracking specific marketing campaigns via additional parameters in URLs, compare `https://support.google.com/analytics/answer/1033863`, retrieved 15 November 2021.

order, this is another business case. Optimizing the entry to the checkout funnel may lead to a higher conversion regarding orders afterwards.

Aside from tempting users to visit the listed shops or to place an order, the ability to contact users is another important business case. The more users are reachable via customer relationship management (CRM), the more traffic is likely to be routed directly to the website without additional steps like using a search engine. The traffic will again result in lead-outs and orders, fulfilling the original business cases. Considering the direct nature of this traffic, it is much more valuable than users coming from other channels. The assumption is that having a direct contact to the website makes users more likely to visit the website more often on their own behalf.

There are different ways to get a user to be available for CRM campaigns. The most obvious point of contact is the price alert, as it is prominently placed on the product page. It has the largest reach not only because of the usability and popularity of the feature itself, but also due to its positioning. By requesting a price alert, the user sets up an idealo account and agrees to being contacted via the newsletter. The newsletter consists of different campaigns maintained by the CRM team. With the opt-in given by the user, they can then send personalized emails.

The idealo account itself does not necessarily grant the ability to contact a user via email, but is a business case nonetheless. It allows for better personalization and understanding of the user, because the user gets another layer of identification by registering with an email address. The additional possibilities to interact with the website, for example, the wish list and price alert features, will make a registered user more likely to enter into the pool of contactable users.

Using the accounts, idealo can then legally establish a more reliable user definition compared to using cookie values for identification. This helps in tracking the behaviour of users over longer time periods. Being able to improve tracking of account users is an advantage in evaluating the moneymaking business cases, therefore the idealo accounts are a business case in their own right. The account potentially enables the evaluation of user activity across different devices, which makes the ability to contact users another important business case.

3.4 Overview of Tracked Content

The following section will give an overview of the various tracked contents and introduce the terminology used in the remainder of this dissertation. As not all tracked information is important for the conducted experiments, only essential fields are explained. Fields in the log files are equal to the columns in the dataset that is created from the log files. See Table A1 for a complete set of columns in the dataset. From now on, column names will be written in bold lower-case type and specific values in these columns will be written in italic font. Any technical terms describing measures or specific variables will be written bold as well.

To understand what leads to certain user behaviour in relation to the business cases, the tracked files contain information about the content the user has seen or interacted with in addition to fields relevant for constructing a session. Every page has its own set of content reflected in the data. The content of every trace can therefore be divided into session attributes and content attributes, alongside technical information used for constructing sessions and valid traces. The technical information in the log files is relatively simple and straightforward. Every trace in the data has the following fields: **cookie_value**, the **tracetime** of the trace, the **http_user_agent**, **page_type** and **page_template**, **url** and **http_referer**.

The **cookie_value** is set by the browser and is used as an identifier for browsers and devices. **Page_type** and **page_template** define the contents and type of a page. **Page_type** can have the following values: *pageimpression* for page impressions, *leadout* for lead-outs or *useraction* for user actions, representing the type of interaction a user performs with the page. Associated with this is the **page_template**. While **url** and **http_referer** refer to specific sites, the **page_template** describes the general type of a page. Every site-type on idealo has its own name in the **page_template** and a set of related content attributes, which are connected to the respective template.

Following the category tree, the homepage *www.idealo.de* has the **page_template** *MainProductCategory*, as it is the index for all attached root and subcategories. Content-wise, there is nothing tracked specifically related to this template. It is not clear what the user has seen – the data does not show what kind of teaser is online, what kind of recommendations the user gets nor how the user proceeds to interact. The last point is of particular importance. Despite having **url** and **http_referer** available in every trace, it is not clear how exactly the user went from one page to another. For example, coming from the homepage to a root category is possible by either clicking on the link in the category navigation or by clicking on 'suggest' in the search bar when typing the name of the category. The difference is not clear in the data.

Every level beneath the homepage without associated products is tracked as the **page_template** *SubProductCategory*. This means that there is no differentiation in the data between the root and subcategories shown in the category tree in Figure 3.1. To differentiate this, the associated **category_id** can be used. Every category – be it a root, a sub- or a product category – has an associated **category_id**. Like the homepage, the **page_template** does not contain any tracked information about shown content. Only the lowest level of categories, the product categories, contain content information.

The associated **page_template** *ProductCategory* as well as the *FreeProductCategory* for categories that promote shop offers only, do contain details about what the user has seen. The main focus here is to know the number of items the user can see. During the time of the page impression on these pages, the number of products, offers and clusters on this page is tracked. This is supplemented by information about the overall number of these items in the category, which is a number also displayed to the user at the top of the page.

The search result **page_template** is called *MainSearchProductCategory*. Alongside the **query** field, which is extracted from the **url** the same way as the sorting parameters, the same content is tracked as in the *ProductCategory*. In the **url**, the **query** terms are attached with a *q* parameter. The search engine is set to redirect unambiguous queries that match a product category directly to the respective product category. This redirect results in a *pageimpression* for the search representing the interaction and the actual *pageimpression* for the product category the user lands on. In these cases, the **url** is supplemented with an additional *qd* parameter.

Further down the category tree are the product pages with the **page_template** *OffersOfProduct*. Being the most important page, naturally a lot of content is tracked here. Most of the tracked fields are related to the prominent list of shop offers, as this is the core of the business. For the first 10 offers, the logs contain information about whether the offer is available for a *Direktkauf*, the speed of delivery when ordered (the distinction is made between low, medium and high), the possible payment options, information about whether the offer has the lowest price in the list, what price it was previously and which shop it is from. The data only provides information about the first 10 offers as these are assumed to be visible in a page impression where the user does not scroll down. Besides information about offers, several fields relate the product to the product page, these are: **manufacturer**, **manufacturer_id**, **product_name** and the **product_id** as well as the **product_type**.

The other pages are somewhat smaller in scope in terms of tracking. The list pages with **page_template** *List* are comparable to the search result pages. Same as there, all information regarding the pagination, the quantity of items shown and items overall, are tracked in the same fields. The keyword used for creating the page can be found as a parameter in the **url**. The cluster pages with **page_template** *Cluster* resemble the product page in terms of tracked content.

The other, less important pages mostly do not track additional useful content. The remaining **page_templates** do not hold specific information, but may have parameters hidden in the **url**. These parameters are, when deemed useful, extracted later. An example would be the **page_templates** *Shop*, *Manufacturer*, *SearchManufacturer* and *SearchShop*. The first two are for the informational pages about shops and manufacturers and include their respective identifiers (ids) – **shop_id** and **manufacturer_id** – as parameters in the **url**. The search result pages for manufacturer and shops have the query as a parameter in the **url**[9]. In general, the **url** may hold additional information which will be added to the dataset if possible and viable. This summarizes the tracked content for the *pageimpressions* on the most important **page_templates**. *Leadouts*, on the other hand, always have the **page_template** *GoToShop*.

[9] Both pages ceased to exist in 2021.

3.5 Issues when Working with Data

Despite having a relatively consistent data model, there are a lot of issues with the tracking and the data. These issues should always be kept in mind as they relate to many aspects and can massively impact the interpretation of results. Some of these problems relate to the design of the tracking set-up itself, others are caused by the natural development of the portal and technology in general.

Beginning with the tracking system, one of the most important issues is the utter lack of tracking in the areas accessible via login. Once a user enters the pages behind the login, any session model using a path with a matching **url** and **http_referer** loses its foundation. This leads to an inconsistency, as a new session begins as soon as the user enters a tracked page again. It might then be the case that significant behavioural markers on the page get lost. The problem of breaking sessions is not caused by untracked areas alone. Technological improvements to browsers, such as plugins that prevent advertising, scripts and therefore tracking, or even the users that enable the 'no-cookie' setting in their browsers, all of these lead to the creation of a new cookie for every new trace. The same is applicable to users who opt out of being tracked – which is a preference introduced in May 2018 under the General Data Protection Regulation (GDPR). The new cookie per trace leads to the generation of a new session per trace as they cannot be connected any longer. Users who behave in this way are essentially useless to the behavioural analytics as their behaviour is impossible to understand in any way. The logs generated by these users are simply not connectable.

Using the cookie as the identifier (id) for session generation is the foundation for a lot of potential problems. Changing or neglecting a cookie using technological means is easy. Furthermore, the cookie defines a device or a browser, not a user. This means, there is no failsafe way to identify an individual user. There is no traditional user entity, instead the cookie is the most reliable identifier despite all the associated problems. The missing user entity leads to even more interpretation problems when thinking about using different devices for browsing or shared devices by different users. Multiple users using one device cannot be differentiated and multiple devices used by one user cannot be easily connected.

A way to circumvent this in cases where the user has an idealo account is the email hash. If a cookie visits idealo while being logged in, the tracker logs a hashed version of the email address used for registration. By associating this hash with different cookies, a single **user_id** can be generated. The problem here is that being logged in is not required. When not logged in, the hash is not tracked. A new cookie cannot not be associated then because the hash is not tracked. Another problem with this approach arises again when multiple users interact on the same device. When these users have different accounts, different hashes get associated with one cookie, leading again to inconsistencies.

Another important point to consider is the user's consent to tracking. If a user opts out of being tracked, all associated traces will not be tracked at all, except legitimate interest data that will be tracked but won't be used for analysis peruses. This leads to these users being completely absent from the data.

When looking at the data, the problem of missing user identification is the focal point of issues with the data and modelling a concept of sessions or journeys. Not having a **user_id** means that the **cookie_value** is the most reliable identifier in the data, alongside the hash. For the analysis of different modelling approaches, this must be kept in mind. To be able to model a consistent model in the next chapters, only returning cookies are used. Other inconsistencies occur over time. There are a variety of issues:

- Bugs in the tracking
- New features or products
- Design changes
- Incorrect modelling and logical inconsistencies

All of these issues may have a huge impact on the data quality. Bugs affecting the tracking may result in missing fields, incorrectly filled out content or, worse, completely missing traces. Missing content can lead to incorrect interpretations and inaccurate assumptions. When the interactions of a user with a specific field are analysed over time and at some point the content is missing, the analysis is more than likely going to be spoiled. For accurate analysis, these cases have to be excluded from further research. The same is true for completely missing traces. Here, it is even worse, as the user behaviour cannot be analysed at all, even at a higher level.

Changes to features, adding new products as in new pages, or even design changes, can result in inconsistencies as well. These cases are often not as dramatic as bugs because the change in the data is planned. Often, it is a case of information being added rather than missing content traces. Depending on the case, this can be easily modelled into the data, adding more value. New features need to be implemented in a reasonable way, however, as it does not make sense to compare datasets regarding a certain behaviour when any feature related to this behaviour differs.

As these pitfalls are commonplace, the modelling logic needs to be carefully developed and checked to avoid inconsistent logics. An example in the data used here is the HTTP status for the different interactions – a valid page impression always has the (HTTP) **status** *200* while a lead-out always has the **status** *301*. The different properties of the file contents should always be kept in mind.

Another issue in the available data is the state of app tracking. As user interaction with the mobile app differs vastly to their interaction with a website, the tracking is also very different, not only from a technical perspective but also content-wise. As of now, the quality of the tracking on the app does not meet the standard of the web-based tracking. Therefore, and to avoid further data inconsistencies, this dissertation does not include data tracked directly in the app.

Chapter 4

Research Design

This chapter describes the steps involved in modelling mechanical and logical sessions in log files, how they are compared and evaluated, and then discusses their impact on different applications. The first section of the dissertation provides an overview of the definitions and concepts used and experimented with. The next section explains the architecture used for working with the data. Afterwards, all steps to produce a working dataset are described, along with and a brief description of the dataset. The following parts explain the identification of different types of sessions. All approaches are explained and adapted to work with the dataset at hand. The final part will outline the methods used for evaluating the various approaches that were implemented.

4.1 Concepts and Definitions

To better understand the experiments conducted in this dissertation, a few terms and concepts need to be clearly defined. As already mentioned in Section 2.2, there is no clear terminology or group of definitions. This section, therefore, discusses the most frequently used vocabulary and tries to consolidate the terms for different concepts more precisely.

The first concept to be discussed is the term user. To group interactions into meaningful segments, some kind of identifier (id) is required. Interactions are then assigned to such an identifier, usually representing a user, a device or a browser. In the early stages of research on session detection, identifiers could be easily found: for instance, in the library catalogue, users usually logged in and out when using it, thereby generating a unique user-id along with a unique, temporary visit-id [120]. This allowed systems to identify users over time and across multiple visits. Under these circumstances, such an id represents an individual user and may be assumed to represent the human being behind the id.

Often though, this is not the case. When an actual user identifier is not available, there are a limited number of other options. Earlier works dealt with finding the best approach to grouping events to a single user without assigning any specific id [95, 232], often resulting in relying on IP addresses[1] for grouping.

[1] `https://en.wikipedia.org/wiki/IP_address`, retrieved 1 December 2021.

Newer and more sophisticated methods for identifying users are called fingerprinting, representing another research direction [87]. The term refers to the identification of unique users without any reliance on the existence of an identifier. Fingerprinting as a term covers a variety of different approaches. To identify a user without a reliable identifier, these approaches utilize transmission control protocol (TCP)/IP addresses [21], different hardware or operating system (OS) features [8, 38] or even more abstract features like font settings or screen resolution specifics [25] to infer highly specific feature sets that allow the identification of a unique user. Nikiforakis et al. [195] loosely classify fingerprinting methods according to the features they use, resulting in categories like browser customizations, browser family and version, OS or hardware and network.

The other option is to use an identifier set by the browser – the HTTP cookie[2]. A set cookie is unique to every browser (so long as it is not deleted nor renewed), but only in theory is it unique to every user – users could share devices and browsers [232]. Therefore, using a cookie, referred to as **cookie_value** in the dataset, brings with it some limitations. It may lead to inaccuracies regarding the mapping of interactions to unique users. Also, some browsers support a functionality that enables the removal of cookies after every click (or browser opening). Such a functionality can result in many **cookie_values** with only a single interaction. Since fingerprinting brings a number of new implications regarding security and privacy concerns[3], this research focuses on the use of a browser id by using the **cookie_value** set by idealo[4].

This means, the term user is actually misleading when talking about individual entities performing interactions. In practice, these are not individual human users but only devices, or more precisely browsers that may be used by multiple persons. For the sake of clarity, this dissertation will nonetheless refer to **cookie_values** as users by applying the concept of a **user_id**. Clearly, there are limitations and potential errors, but the assumption of a **cookie_value** (or multiple **cookie_values** depending on the applied **user_id**) equalling a user by connecting multiple cookies when they share an email address is valid enough for the context of session definitions. As long as a cookie and therefore a **user_id** performs multiple interactions, the assumption of one user may hold in the context of this dissertation's hypothesis. Details about the implementation of the **user_id** are provided in Section 4.3. For now, the definition is as follows:

Definition 1 (User)**.** *A user is an entity performing interactions with a given system.*

This analysis restricts itself to **user_ids** that perform at least two interactions. Other **user_ids** are filtered out using multiple conditions during preprocessing as explained in Section 4.3. This restriction is put in place because it is not possible to detect different session concepts with only one interaction.

[2] `https://en.wikipedia.org/wiki/HTTP_cookie`, retrieved 15 November 2021.

[3] Fingerprinting largely ignores any consent (given or declined) to tracking, making it a somewhat questionable practice. Compare `https://www.nytimes.com/2019/07/03/technology/personaltech/fingerprinting-track-devices-what-to-do.html`, retrieved 15 November 2021.

[4] The **cookie_value** set here is not affected by a server timeout and can be understood as an identifier that is permanent until the user deletes it.

Summarized, the term ‘user’ equals individual **user_ids** in the data so long as they perform at least two interactions despite not necessarily representing an individual human user. Having now established an abstract concept of a user who performs interactions with a given system, the logical next step is to define what an interaction and the related concepts are.

Definition 2 (Interaction)**.** *An interaction describes any tracked form of action a user performs on or with a given system.*

The tracked form of action is also called an event. Event and interaction are therefore interchangeable. In the use case at hand, an interaction is either a page impression, a lead-out or a user action. As explained earlier, user actions are the actions a user can perform on a specific page, for example, adding a product to the wish list. Page impressions are basically every click on a page on the website – the concept indicates that the user gets an actual impression of the page, viewing its contents. Lead-outs are the clicks on any link directing the user away from the page. These types are referred to as *pageimpression*, *leadout* and *useraction* in the data.

Interactions may belong to multiple overarching concepts that could consist of one or multiple tracked events. The most straightforward and fundamental concept is a visit.

Definition 3 (Visit)**.** *A visit is an unspecified number of subsequent (connected) interactions made by an individual user with a given system with exactly one entry point and one known last interaction before a new entry point without considering other possible boundaries.*

An entry point or lead-in can be any external source leading to the website: a search engine, a bookmark or even a shared link. The known last interaction defines the end of the said visit; a new entry point indicates the start of a new visit. The lack of possible boundaries means that a visit is not separated by any form of timeout; in theory, it could last days. Naturally, closing a page would also end a visit although this is not tracked by the system and therefore purely hypothetical.

The definition becomes fuzzy when multiple entries are considered at the same time. Theoretically, a user could enter a website on many tabs using differing entry points. Since they have differing entry points, these are considered different visits although they could theoretically end on the same page or content. Considering this, a visit is inherently detached from an information need in this dissertation; a visit may pursue one or many information needs, but the term first and foremost represents the (mechanical) interaction with a system.

In the literature, the concept of a visit is equal to the concept of path-based mechanical sessions as was referenced in Section 2.3.2. Often, visits are assumed to be connected with pursuing an information need. Indeed, attempts have been made to replicate the concept of a visit by session-detection approaches regarding the pursuit of information needs. This dissertation does not inherit these assumptions – a visit is a purely mechanical construct replicating the single interaction or subsequently performed group of interactions with a system.

Information needs come into play when using session approaches, which evolve around the concept of a visit. Following the definition of Manning et al. [170], an information need refers to a topic a user wants to know more about. This definition is focused on the use of search engines by using queries to fulfil an information need. Considering the e-commerce data used in this dissertation, the desire for knowledge about a topic may be too narrow. Therefore, an adapted version of the definition is used:

Definition 4 (Information Need)**.** *An information need represents a topic that a user wants to interact with, in the way a given system is enabled to allow.*

With this definition, all kinds of interactions with the system by the user are covered. The information need here is also limited by the given system, for example, usually, a user would not be able to check for the weather on a price comparison site. For the use case at hand, any information need is more or less associated with a category (e.g. interacting with the category page for *Smartphones*).

When dealing with the fulfilment of an information need (i.e. interacting with a **category_id** or issuing a **query**), the predominant term used is 'session'. As presented in Section 2.2, the range of definitions for the term 'session' is broad, but they usually revolve around the same foundation: having one or more interactions that are related to fulfilling an information need. The differences originate from the conditions that define the boundaries of such a session. Usually, this is a temporal inactivity condition trying to emulate the start and end of a series of information need-related interactions. In this dissertation, the term session is defined as an overarching and general concept.

Definition 5 (Session)**.** *A session is a series of interactions made by an individual user with a given system in order to fulfil one or multiple related information needs.*

Initially, there is no boundary involved here. The term session only describes (tracked) interactions with a system in an attempt to fulfil an information need – regardless of other boundaries. This could involve several visits, a single visit or even multiple parts of multiple visits; sessions are the abstract counterpart to the precisely defined visits.

Possible boundaries are introduced in more specific concepts. These deserve their own definitions since the type of boundary may have a strong impact on how the respective session concept deals with the representation of the fulfilment of information needs. These definitions are based in part on the concepts explained in Section 2.3. The main distinction in this dissertation is made between mechanical sessions and logical sessions.

Definition 6 (Mechanical Session)**.** *A mechanical session is a series of subsequent interactions made by an individual user with a given system delimited by a mechanical boundary assumed to fulfil an information need.*

A mechanical boundary could be a temporal inactivity timeout or a maximum number of interactions or even a maximum duration starting from the first event. Mechanical means that there is a hard boundary with no (inherent) topical connection between subsequent events. The difference between this and the overarching term is that a mechanical session is assumed by definition to focus on fulfilling one information need. This is true for temporal sessions (i.e. mechanical sessions with a temporal boundary) as well as for lexical sessions (i.e. mechanical sessions delimited by lexical similarity) – both variants only work with that base assumption, although lexical sessions may gradually focus on broad and narrow information needs, depending on the implementation.

A special case is seen in the structural-based or path-based methods using the relationship between **url** and **http_referer** to reproduce the path an individual user takes on the website; this type of mechanical session equals the concept of a visit from definition 3 since visits are not assumed to fulfil an information need, but rather only replicate user behaviour.

Next to mechanical sessions are logical sessions. Here, no obvious hard boundary indicates the end of a session. Also, even more so than in typical variants of the mechanical session, a logical session may span multiple visits and is not limited to subsequent interactions.

Definition 7 (Logical Session)**.** *A logical session is a series of interactions made by an individual user with a given system in order to fulfil an information need delimited by topical constraints.*

The topical constraint is the main difference to a mechanical session, although boundaries become blurred when comparing this type of session with lexical sessions. Where lexical sessions as part of mechanical sessions in most cases have an explicit ending due to their nature, logical sessions may continue until an information need is fulfilled. This gets even more abstract considering the following examples. In a visit with three interactions, there could be three logical sessions that may or may not be continued in the next or another visit. As a general example, these three logical interactions could be checking the weather, reading emails and looking at news on a mail-provider website. Logical sessions are also often referred to as tasks, which is a more figurative way of describing the concept: a user has a task to fulfil a specific information need, which may consist of one or multiple related interactions that are not necessarily subsequent. The term task is not applicable to mechanical sessions as the information need is usually not visible there (except for lexical sessions, which are more of a mixed form).

Logical and mechanical sessions are focused on fulfilling a concrete information need. Broader information needs like planning a vacation may fit into the concept of logical sessions, but the literature sees these as a separate research problem named complex search tasks. Complex search tasks describe a broader problem requiring multiple smaller subsets of information needs that are not necessarily directly related. The example of planning a vacation is actually a very reasonable one. Booking a flight and a hotel and planning individual day trips to different places of interest are technically different information needs and may also contain different search terms and definitely different contents.

A separate definition is therefore appropriate. Since the label complex search task is heavily focused on the use of a search engine, this dissertation refrains from using the term complex search task and instead introduces the term journey. This term is more fitting in the context of the analysed data, because fulfilling the information needs here does not necessarily involve searching – as in querying – a search engine.

Definition 8 (Journey)**.** *A journey is a set of related logical sessions made by an individual user in order to fulfil multiple related information needs that belong to a more complex information need.*

Considering the context of the case study at hand, a journey may involve every information need revolving around a bigger purchase. This could be, for example, the construction of a computer – buying separate parts like motherboard, CPU (central processing unit) and GPU (graphics processing unit) may be somewhat similar, but ultimately, they are different (but related) information needs.

The focus on logical sessions is ultimately necessary since the basic assumption is different and more advanced compared to mechanical sessions. In theory, mechanical sessions could be strung together to form a journey, but the link between them would not be as clear and comprehensible as in logical sessions. Mechanical sessions are not technically restricted to the same information need, although the basic assumption assumes that they are: they are assumed to be self-contained constructs. Therefore, they are practically excluded from journeys. Logical sessions, in contrast, could range across multiple visits to link interactions on the same information need over time, and to connect different interactions using that same information need.

Figure 4.1 (on the next page) shows a practical example to illustrate and clarify the different relationships of the defined terms and their variants. The figure depicts a user performing multiple visits, showing how the different interactions could be grouped with regards to the various concepts.

As can be seen in the figure, the user performs multiple interactions. The smallest circles each represent one interaction, for example page impressions (PI), queries (Q) or lead-outs (L). Small circles in close proximity belong to the same visit, resulting in five visits in total as can be seen on the lowest level. These five visits are divided into six mechanical sessions by, for example, a temporal inactivity timeout between the interactions, indicated in the second lowest level. Mechanical sessions 2 and 3 divide visit 2 into two separate units. The bigger circles around the smaller circles indicate logical sessions. These may span multiple visits, which do not have to be subsequent. Both logical sessions may deal with a different information need, although the overarching journey at the highest level connecting them implies these different needs are related.

The key difference between a logical session and a mechanical session is therefore mainly derived from the handling of the information need. A logical session may span multiple visits focusing on exactly one known information need. ‘Known’ means that it is clear that the logical session is supposed to fulfil only this one information need – there is no multitasking regarding the information need involved here, as all interactions originate

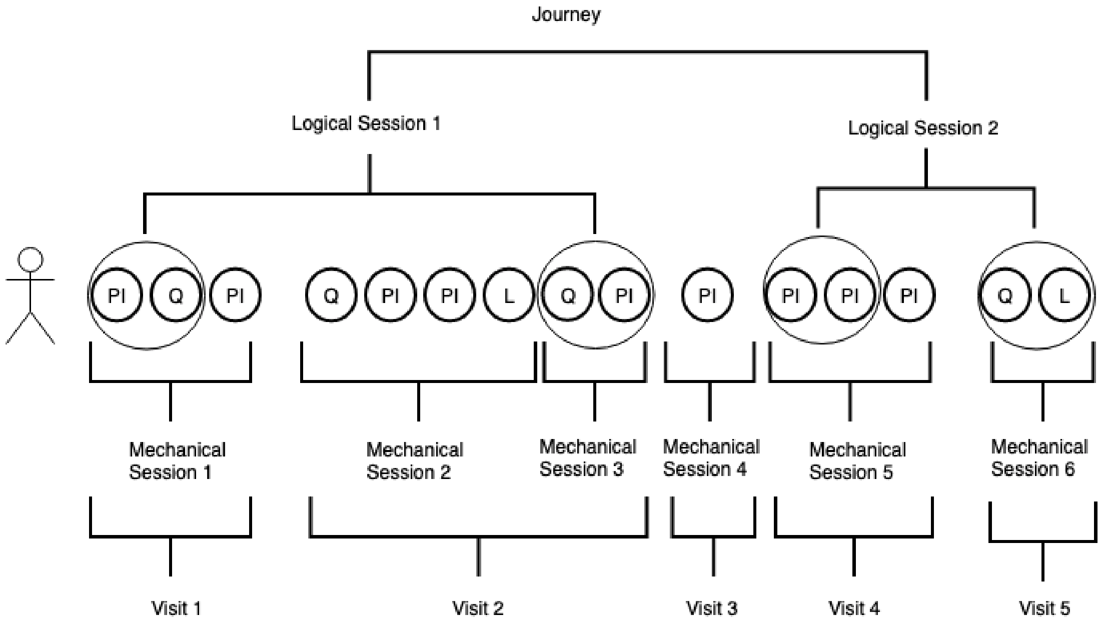


Figure 4.1: Schema of session concepts. Abbreviations: PI = Page Impression, Q = Query, L = Leadout. Small circles represent a single interaction, bigger circles indicate a topical connection between the interactions.

from the same intent. A mechanical session has a different base assumption: by setting a mechanical boundary, it is only assumed that one information need is worked on within the limits set by this boundary, but the segmentation is not defined by this assumed information need. This dissertation tests both mechanical and logical sessions as well as a combination of them to estimate the impact of each of these approaches on different use cases, following the hypothesis that logical sessions, with regard to the actual context of a user, improve the outcome in a variety of settings.

4.2 Architecture

This section shows the infrastructure used for preprocessing and modelling. The tracking component is not part of this dissertation, therefore, it will not be described in any greater detail than the description given in Section 3.3. The extract, transform, load (ETL) pipeline presented starts after the data is logged and transformed. Bot filters are already applied and any logs irrelevant to user behaviour such as statistic pixels or dynamically loaded contents are removed. The different steps are reflected in Figure 4.2.

The logging system that collects all relevant logs and processes them is called Splunk[5]. Here, all logs are logged in the form of individual JSON files that represent the clicks a user makes on the website. Since the system is limited in its capacity and calculating power, the first decision was to extract the relevant information from the logged data and push the resulting files to the cloud.

[5] `https://www.splunk.com`, retrieved 15 November 2021

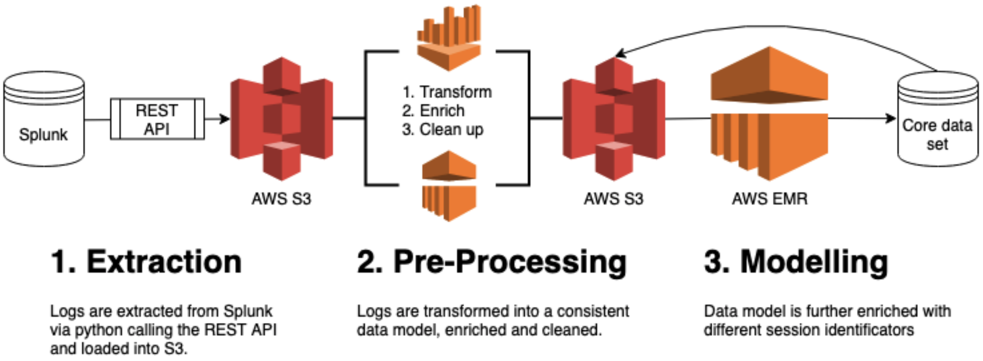

Splunk
REST API
AWS S3
1. Transform
2. Enrich
3. Clean up
AWS S3
AWS EMR
Core data set
1. Extraction
Logs are extracted from Splunk via python calling the REST API and loaded into S3.
2. Pre-Processing
Logs are transformed into a consistent data model, enriched and cleaned.
3. Modelling
Data model is further enriched with different session identificators

Figure 4.2: Workflow of preprocessing architecture.

To achieve this, a Python script looped through every hour of every day of 2018 and issued a query to the Splunk system. The results were uploaded to a dedicated S3 bucket. S3 refers to the object-based cloud storage service, known as simple storage system[6], provided by Amazon Web Services (AWS)[7]. The hourly processing was required considering the quantity of data that is transferred. Splunk is not designed to handle such huge amounts of data at once, therefore every hour was processed separately. Having the data on AWS Simple Storage System (S3) allowed use of Amazon's management services. Here, a bucket structure was created for the data, containing the raw processed data as well as versions of all processed steps.

For all further processing, in the main AWS Athena[8] and a custom-designed elastic map reduce hub were used. AWS Athena is a query engine based on Presto[9], a distributed query engine designed for big data. Athena is conceptualized for directly accessing files on S3, allowing access to the contents of text files via Structured Query Language (SQL). Since it supports 'create table as select' (CTAS) statements and inserts, Athena is a viable and cost-effective option for less complicated processing steps depending on the use case.

For the heavy lifting, Amazon's Elastic MapReduce (EMR)[10] service was used. EMR is a service providing several big data frameworks like Spark or Presto on dedicated server clusters. To utilize this, a Jupyter[11] hub was created that automatically allows easy access to Jupyter notebooks with a pre-installed Spark framework when the cluster is booted up. Further preprocessing was then done by using PySpark and Spark SQL. For accessing the processed data, different cluster configurations were used with regards to the task at hand.

[6] `https://aws.amazon.com/de/s3/`, retrieved 15 November 2021.
[7] `https://aws.amazon.com/`, retrieved 15 November 2021
[8] `https://aws.amazon.com/de/athena/`, retrieved 15 November 2021.
[9] `https://prestodb.io/`, retrieved 15 November 2021.
[10] `https://aws.amazon.com/de/emr/`, retrieved 15 November 2021.
[11] `https://jupyter.org/`, retrieved 15 November 2021.

4.3 Creating the Dataset

This section explains how the aforementioned architecture is utilized to run different queries and preprocessing methods. It follows the same structure as the previous section and describes how the log files are extracted, loaded to a cloud storage and processed afterwards.

Step 1: Extraction

The query used for extracting the data in step one basically collects all relevant data for the duration of one hour along with making some minor adjustments to the content. The query is simple: it selects all relevant data from the Splunk index[12], removes possible bots and all interactions that are not directly associated with interactions on the website.

The process is carried out by utilizing the fields **botstatus** and **page_type** from the log files, removing the bots, and with regard to the **page_types**, only selecting the event-types *pageimpression* (representing a page impression), *leadout* (representing the lead-out to an external website) or *useraction* (representing an action by the user), since only these represent valid interactions. The (HTTP) **status** is also checked to only include fully loaded events with the value *200* (HTTP status 'OK') in the case of a *pageimpression* or redirected events with **status** *301* (HTTP status 'Moved Permanently') in the case of a *leadout.* In the end, every field is grouped by at least the **cookie_value** and the **tracetime**, which is the timestamp of an event in unix format. At this level of granularity, all events are assumed to be unique and are treated as such.

To create a unified and anonymous **user_id** later on, the **cookie_value** is enriched with an associated hashed **email_id** from an internal database, where every registered email address hash is mapped to all **cookie_values** that were seen at some point in the events. This internal table only takes into account users who agreed to opt-in. All encounters are saved in N-to-N relationships. For the mapping, it is assumed that all **cookie_values** belong to the latest saved email hash. This may introduce errors for multiple users surfing on the same device. Since the presented user concept is sufficiently abstract, using this method should be viable. For now, every **cookie_value** gets associated with a numeric id for the latest seen email hash in the data. The resulting file is saved and uploaded to S3.

It is important to note that all data used in this research is tracked under the conditions to which users consented and is operated under a strict agreement between the researcher and idealo (as the information system analysed in the use case) on the usage of the data. This protocol has been observed to ensure that the data analysed herein does not violate any user-privacy regulations and, furthermore, that user wishes regards their agreement to be tracked have been respected.

Step 2: Preprocessing

Once the dataset was available in the AWS S3 storage system, the actual preprocessing could be conducted, depicted in step two of Figure 4.2. The easier calculations were done

[12] A Splunk index can be understood as a database table in which every log file represents a row in the table.

directly via SQL on the AWS Athena query service. First, as a necessary step to enable Athena to correctly read the data types and the partitioning, a table was created on top of the files. Since the easier steps are calculable without having to process all data at once, a parallelized SQL day-per-day approach seemed to be a fair choice. For this to work, a Python script was created to loop through the necessary parameters (e.g. date boundaries) and run a query for every combination of parameters, saving the results back to S3.

The first action entailed a clean-up step: to achieve a consistent data model, the so-called checkout funnel was aggregated to become one event. The checkout funnel is the process of initiating a purchase directly on the website by clicking on the blue button on a product page. It should be noted that the differences in the number of events in that funnel, which are due to changes in the tracking over the time period of the dataset, would have influenced later analysis. Since the checkout logically represents only one action (making a purchase or not making a purchase), the reduction to one event is reasonable even though some information may be lost in the process. By simultaneously joining the information of a successful order to the newly created event, the interaction may represent a *leadout* with additional information about the user's success in ordering an item. These events are saved in a separate table, which is then combined with the remaining base data. First, the base data is selected and enriched with additional information. Basically, this step consists of several joins with lookup tables to gather further information, as well as a lot of case statements with regular expressions on the fields **url** and **http_referer** to extract any missing identifiers or variables directly from them.

One of the more important steps to ensure consistency of the data model is to make sure that all important identifiers are actually present in the data when needed. Unfortunately, this is not inherently the case. In a great number of events, the respective identifier for products or categories are not tracked by default. Fortunately, most of these events do contain these identifiers in the URL, which is why the field **url** is parsed for **product_ids** on many different pages. The same is done for the field **category_id**.

Additionally, information about the category hierarchy is joined using a lookup containing the **category_id** and the respective **category_type**, **parent_category_id** according to the category tree as well as the **category_name** and potential **category_synonyms**. Another lookup joins product information including the **main_product_id**, the **product_name**, the **product_type** and the **manufacturer_id** to every **product_id**. The **main_product_id** is attached to every variant product. All products will have their own individual **product_id** in this field. Additional **category_id** data is joined by utilizing internal idealo inventory databases to get **category_ids** for **cluster_ids**.

Using the **url** and the **http_referer** fields, **query** parameters are extracted. The general query parameter is indicated by a *q*. Queries that are identified as specific category names by an internal system are indicated by a *qd*. All terms are collected in the field **query** and **referer_query** respectively. All queries are normalized by removing all special characters, any trailing blank spaces from both sides, and reducing the number of all remaining characters. Both fields are capped at a string length of 100 characters for performance reasons. Queries with less than 100 characters make up 99.9% of all queries.

Additionally, any search terms from the **page_types** *SearchShop* and *SearchManufacturer* are stored in the fields **shop_query** and **manufacturer_query** by using regular expressions on the URLs on these pages. *List* pages also contain a query, stored in **list_query**. For these queries, the **http_referers** are also parsed for the respective referer queries.

As another step, a numeric **user_id** was created and joined to every row in the table. This **user_id** was calculated via the **email_id**, which was joined in from the internal mapping table. If multiple **cookie_values** are related to an **email_id**, all related events get the same overarching **user_id** (which is basically just a substitute for the originally applied **email_id**). If there is no **email_id**, all events related to the same **cookie_value** get the same **user_id**, which for now is just the **cookie_value** as a string. The mapping between **cookie_value**, **email_id** and **user_id** is removed afterwards so that a connection is no longer possible.

The next step created a lookup for these **user_ids**. The responsible query sums up all *pageimpressions*, *leadouts* and *useractions* and groups them by the newly created **user_ids**. This is done in one query over the complete dataset since the overall amount is needed. Afterwards, a simple rank function generates a unique integer for every row. The result is a general, numeric **user_id** that can now be joined back. After joining the **number_of_interactions** and the general **user_id** to the original dataset, the sum total of interactions can then be filtered based on the overall number of interactions from every **user_id**.

The following steps were then performed via Spark SQL on an EMR cluster, because work was required on the dataset as a whole at the same time. Several tasks were performed. At first, the **cookie_value** was removed from the dataset to meet GDPR compliance[13]. Another task was to calculate the **timespan** between all events from the same **user_id**. For this, each **tracetime** from the event is dragged by a window function from the event directly before the reference event. The **tracetime** of the **last_event** is then subtracted from the **tracetime** of the reference event. Since every timestamp is in unix format and milliseconds, the resulting calculation is then divided by factors 1,000 and 60 to achieve the **timespan** in minutes.

Having performed the first important step regarding the internal structure of the dataset with this query, the next step could carried out. Using the **number_of_interactions** from the previously joined **user_id** lookup, all **user_ids** with less than two interactions were removed. Less than two interactions mean that the respective id has exactly one *pageimpression* or one *leadout* or one *useraction* associated with it. This is reflected as a single row in the dataset – there are only these three types of interactions. There is no way of identifying different session concepts here since there is only one interaction overall. By removing these **user_ids**, the number of unique **user_ids** in the dataset is reduced by around 34.72%. The number of interactions, equalling the number of rows in the dataset, is reduced by only around 3.92%.

[13] Thereby ensuring that no **email_hash** or **cookie_value** can be associated with the **user_id**.

The final preprocessing step is actually closer to the modelling approaches that will be presented in Section 4.4.2.1. By introducing a visit concept to the dataset, it is possible to analyse certain user behaviour patterns. This is necessary to understand how the website is used in a practical and descriptive manner: how often do users visit the website and how many interactions are made in every visit? To answer these questions, a **visit_id** is calculated on a **user_id** basis and added to the dataset. Details for the implementation and potential drawbacks can be found in Section 4.4.2.1, as the visits technically equal a session concept.

Step 3: Filtering

With the between-interaction time in the form of the field **timespan** now available in the data, more steps regarding further preprocessing were possible. To enable this, another table with summary data users was created. This table included all **user_ids** with a minimum of two interactions and the related amount of time spent overall on the website; including the total number of products, categories and manufacturers viewed, the queries and the already-known total number of total interactions. With the next query, three more subsets of users in total were then removed by applying the following rules:

1 overall time spent on site below five minutes, zero queries and zero or one visited manufacturer or category

2 overall time spent on site below five minutes, exactly one query and zero visited manufacturers and categories

3 overall time spent on site below five minutes, one visited manufacturer, category or query and maximum three interactions

The underlying idea is relatively simple: having a five-minute inactivity window as the lowest temporal session boundary will result in all the removed subsets equalling exactly one session. The other conditions ensure that any logical session or journey approach will also result in exactly one segment of interactions. Visiting a maximum of one category or manufacturer marks these interactions as related to the same information need. The same is true for issuing exactly one query without any other interactions whatsoever – like browsing categories or manufacturers. Besides, these interactions are very likely to belong to the same information need. This step reduced the number of users in the already-filtered dataset by another 31.99% and the number of interactions by 7.39%.

In part, the reason for using manufacturers instead of products relates to the structure of idealo's website[14]. Another reason is the high probability that a visit to one manufacturer in one category refers to the same range of products and, therefore, to the same information need. As an example, visiting multiple products in the category *Smartphones* of the manufacturer *Apple* would be considered the same information need. This may not be true for every category, however, since manual classification of this area is commonplace and may result in errors. In general, though, the assumption should hold true for

[14] See the difference between main and variant products in Section 3.2.

the majority of interactions – especially considering the short period of time in which the interactions where made.

Another hard restriction was made on the **media** of all interactions. Since the tracking quality of the *app* data is not satisfactory, the decision was made to remove all *app* traffic. Traffic from the *app* is generally rather incomplete, lacking in consistency and is overall fairly poor in terms of tracked content. Therefore, *app* traffic is not alone in being excluded; the **user_ids** that have performed actions on the *app* are also removed completely from the dataset, including any interactions they have carried out on the web. Doing so ensures consistency in logical session approaches. While this strategy may not be immune to errors, the internal structure of the remaining data is still valid when it is considered that the overall quantity of removed traffic is quite small (4% of all rows).

4.4 Modelling Sessions

This section provides an overview of the various approaches tested in this dissertation. As a first step, some specifics of the dataset will be explained to clarify the fields in use. This is followed by a description of how all the tested approaches were implemented, including technical details of the calculations. In total, four general areas were tested:

- Visits (path-based structural approach)
- Temporal boundaries
- Lexical and semantic similarity
- Combined approaches

While these general areas may appear to only define the nature of the approach, they include multiple methods and mechanics all differing in complexity. A complete overview of all approaches can be found in Table A2. Below, each area is explained in brief, with the underlying assumptions offered once again with respect to the state of research. For clarity, all tested variants are divided into two groups to either represent a mechanical or a logical session construct.

Regards the further implementation of all approaches, it is important to be aware that marketing-related information has been excluded in all session approaches except for the visit concept. Marketing-related information usually comes in the form of UTM parameters derived from the **url** or **http_referer**. In productive e-commerce systems, this type of information is utilized alongside the traditional session-identification approaches, like timeouts, to indicate the start of a new session. When a user enters the website with a new campaign or uses a new marketing entry (i.e. directly via a link or via search engine advertising), this constitutes the beginning of a new session. The reason for this is simply financial: depending on the entry marketing channel, all financial value generated during the assigned session is attributed to the respective marketing channel leading to this session. It follows, therefore, that as the number of sessions a marketing channel generates is generally used to estimate its efficiency, the budgeted expenditure of the channel will also be affected. As the many different attribution models that inform this information are not connected to user behaviour, it will not be used here.

Before going into details of the implementation, some specifics of the data model are explained. This is necessary to be able to implement some of the approaches, because most of the methods presented in Section 2.3 rely on different dataset structures.

4.4.1 Preliminary Considerations on the Dataset

The dataset analysed in this dissertation contains data from the complete year 2018. After preprocessing, it contains 1,268,619,378 interactions from 78,361,923 **user_ids**. Every row represents one interaction, separated into 1,034,058,301 *pageimpressions*, 229,260,133 *leadouts* and 5,300,944 *useractions*. A more detailed statistical exploration is given in Section 5.1. Table A1 provides the tabular schema for the dataset.

The actual data is divided into page impressions, lead-outs and user actions. The majority of interactions comes in the form of page impressions. Basically, each click made while navigating the website is a *pageimpression* whereas every click navigating towards a shop or an order is a *leadout*. This means, every click on a navigational element or a page on the website and every query is a *pageimpression*. Clicks on links to shops, offers or on advertisements are *leadouts*. A *useraction* is somewhat special; these constitute events that relate to an interaction with user-specific elements on the site. They do not represent page changes but actions on the visited page; i.e. putting a product on a wish list or setting an alert. There is no change in content here, a *useraction* just represents additional information on top of a *pageimpression*.

Every row in the dataset has a **user_id**, a timestamp in the form of the **tracetime** and the calculated **timespan**, indicating the time since the last interaction. Essentially, these are the most important fields for any basic session-identification algorithm. The **tracetime** (in combination with the **timespan**) allows a temporal boundary for any **user_id** to be calculated. For any approaches related to the contents of a page, the **url** and **http_referer** as well as the **page_template** are relevant. With the **url** and the **http_referer**, the path a user takes on the website can be reconstructed, essentially replicating the concept of a visit. An important potential source of errors is the fact that the **timespan** does not necessarily equal the time an actual user has spent on the page. It rather just marks the time that passed between two subsequent events of a user, calculated by comparing the timestamps of said events.

The type of page visited is represented by the **page_template**. This field is the classification of the visited page, thereby unambiguously identifying the type of interaction. For example, the **page_template** *GoToShop* is explicitly associated with a *leadout*. The **page_templates** associated with *useractions* actually describe the performed interaction, for example *activate_pricealert*. All remaining **page_templates** are connected with *pageimpressions*; these actually just describe the visited page like *OffersOfProduct* for the product page.

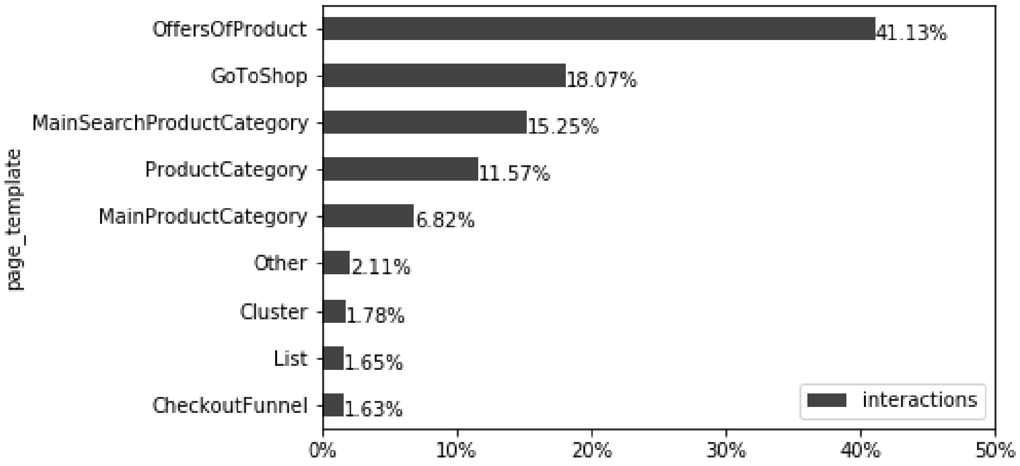


Figure 4.3: Interactions per page_template.

Figure 4.3 is an overview of interactions per **page_template**. Clearly, the majority of interactions happen on the *OffersOfProduct*, which is the product page, and the most important page from a business perspective. The *GoToShop* **page_template** with roughly 18% of rows represent *leadouts*. Of the interactions, 15.25% are search result pages; this means that only 15.25% of the interactions come with a **query** string. Around 11.57% of interactions are made on product category pages with the **page_template** *ProductCategory* and slightly less on the *MainProductCategory*, the homepage. Users interact far less frequently with all other **page_templates** – the *Other* bar is made up of 27 different **page_templates**. Among these are the *SubProductCategory* (0.38%). This is an important indicator of the navigation behaviour on the website: users mainly focus on visiting product pages, either by directly landing on the product pages, using the search or navigating via the homepage and category pages afterwards – there is minimal browsing involved.

Every **page_template** has its own set of identifiers (or human-readable names) that can be used to connect the different **page_templates** to a logical construct. These identifiers are essential, since they represent the content of the page and, in combination with the **page_template**, the type and level of information the user accesses. The most important identifier is probably the **product_id**, associated with multiple **page_templates**: *OffersOfProduct*, *GoToShop* and practically with every *useraction* that involves a product. The latter includes setting a price alert or adding a product to the wish list. They should also appear in the data when a user writes a review about the product using the **page_templates** *ProductRatingForm* and *ProductRatingFormSuccess*. At the same level of abstraction (meaning the lowest level of information) are the *Cluster* and *List* pages. *Cluster* pages have a **cluster_id**, *List* pages come with **list_id**. The *Leadouts* from offers on a cluster page or from search result pages have an **offer_id**. Likewise, almost every **page_template** has a **category_id**, indicating a higher-level topic association.

4.4.2 Modelling Mechanical Sessions

4.4.2.1 Path-based – Visits

The first model (row one in Table A2) to be implemented is a special case, since it is not effectively a traditional session-detection approach as defined in Section 4.1 and according to the definitions set out by this dissertation. The visit as a concept is intended to replicate the actual visit of a user to the site. There are no assumptions on information needs or other boundaries here, it simply represents the subsequent interactions of a **user_id** from the point of entering the site to the point of leaving the site at some point, in one way or another. Visits are supposed to represent a self-contained sequence of interactions with a system; these interactions may belong to the same of different information needs.

The underlying concept resembles a graph of visited **urls** and the corresponding **http_referers** of a **user_id**. For every interaction, the idea is to check if there is a matching **http_referer** in subsequent interactions, and if so, to connect these events. Using this methodology, in an ideal environment, the path of a user through the website – from lead-in to a potential leaving point – could be reconstructed. Since the dataset at hand does not represent an ideal environment, several problems can occur.

Theoretically, interleaving visits could be recreated by comparing **url** and **http_referer**. A possible problem here is if a user visits the same page on two different visits from the same referer – it would then not be possible to distinguish in which visit this sequence of interactions happens. Another thing to keep an eye on regarding these path-based approaches is the quality of tracking. All events need to be tracked consistently to be able to perform proper path analysis between them. If events related to visited pages are missing then a visit break occurs, which potentially introduces a high number of errors. Taking the case of the dataset currently at hand, this is not a potential trivial error, since unnecessary events for other session approaches have also been removed[15]. Another important factor is the absence of tracking in the account area, which may lead to a lot of visit breaks, as described in Section 3.5. Still, it is necessary to implement the concept since it stands as an actually tangible approach to represent user behaviour. A visit describes what the user does without any assumption regarding intents or information needs. As to the quality of tracking, the concept also begs the question of whether the replication of actual tangible user behaviour is even possible.

The concept was implemented using a PySpark approach that iterates through every **user_id** object consisting of all user interactions of the respective user. The algorithm makes a comparison row by row having already sorted all rows chronologically by **tracetime**, and compares the **url** of the reference row with the **http_referer** in subsequent rows. Every time a connection cannot be made, a new **visit_id** is assigned to the row, creating an index of visits for every **user_id**. This includes marketing tracking parameters; their presence in the **url** would also indicate a new visit, as they define a precise new starting point.

[15] For example, *pageimpressions* with **status** *301*.

To reiterate the point: in the approach used in this dissertation, the concept of visits relies exclusively on an existing **url/http_referer** relationship. In the current implementation, no complex special cases are considered that may circumvent any tracking or data-quality issues. The one exception is the connection between identical **url/http_referer** stems that is activated by comparing the path while ignoring additional (internal) parameters. Other cases were ignored; thus, if no relationship is found, a new **visit_id** is set. An example of this is the account area for idealo – if a user visits an untracked account page and comes back to the tracked pages afterwards, a new **visit_id** would be set. While it is acknowledged that issues such as this may suggest a break from the visit concept imposed by this dissertation, to have created a more all-embracing methodology would have set the scope wider than the bounds of the research questions.

4.4.2.2 Temporal – Inactivity

The first actual session-detection algorithm to be implemented follows the (in)famous industry standard. Temporal inactivity (variants are identified with the prefix ti, rows 2–14 in A2) timeout sessions are used widely in all kinds of systems. Companies such as Google use a 30-minute inactivity rule to identify their sessions, although campaign information is frequently used as well to mark beginnings (and endings, respectively) of new visits[16]. A change in the campaign parameter would indicate a change of the marketing channel, which, for financial reasons, is then counted as a new session[17].

The basic underlying assumption of temporal inactivity sessions is that users work on fulfilling one information need before taking a break and then beginning work on another information need. The goal here is to find the optimum temporal threshold to be able to separate the events belonging to different needs. The use of the same general temporal threshold is based on the belief that all users will potentially behave similarly on the same information system. Since the literature is rather undecided about how arbitrary the choice of timeout value can be [51, 83, 184, 192], a broad range of values is tested. In addition, other timeout values were tested to gauge whether alternative arbitrarily chosen boundaries have any impact on the number of identified sessions. Overall, sessions were calculated with 14 different values. The following values (referring to minutes) are used: *5*, *10*, *15*, *20*, *25.5*, *30*, *45*, *60*, *90*, *120*, *180*, *360*, *720* and *1,440*. There are no further conditions, the only boundary used is the time between subsequent interactions. The working method illustrated in Figure 4.4 displays nine interactions grouped into two inactivity sessions. The second session begins because the time gap between interactions 3 and 4 is greater than (or equal to) the inactivity timeout; the time gap between interactions 6 and 7 is smaller, therefore the last six interactions are grouped into one session.

The calculations were carried out using Spark SQL executed from a PySpark script on an EMR cluster. The first part of the query simply checks if the already-calculated

[16] `https://support.google.com/analytics/answer/2731565`, retrieved 8 June 2020.

[17] As explained above, this dissertation has not applied this extra step as it would involve implementing an attribution model for marketing that may be entirely dependent on the respective use case within a productive environment.

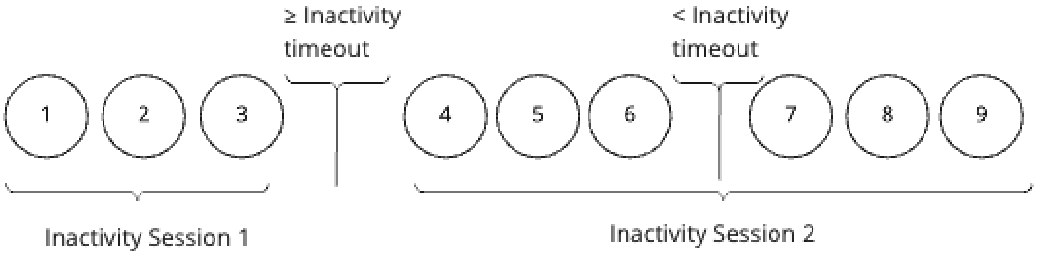


Figure 4.4: Example of the inactivity session method. There are nine interactions grouped into two sessions. The time gap after the first three interactions is bigger than the inactivity timeout (as indicated by the mark), therefore the last six interactions are grouped into the next session. The time between interactions 6 and 7 is greater but still smaller than the inactivity timeout, therefore these interactions are grouped into one session.

timespan in each row is greater than the respective inactivity timeout value. If the value is greater or equal to the timeout value, the value *1* is set, indicating a new session start. Otherwise, the value *0* is set. The next part is a window function that sums the newly calculated session start over the **user_id**. The result is a counter that assigns every session an increasing identifier according to its chronological order. This new identifier is assigned to every interaction belonging to the same session.

4.4.2.3 Temporal – Fixed-Length

The next variant of mechanical approaches are sessions with a fixed duration (variants are identified with the prefix tf, rows 15–28 in Table A2). Using a total threshold on page-stay time is rather old-fashioned and is rarely used nowadays. The basic assumption is that, depending on the contents of the website, a user only stays for a certain time per session, leading to a general overall threshold [238]. This line of thought may hold true for websites with a very limited scope or a limited set of pages with specific purposes, where the particular online business model may be perceived to lead to potentially very short sessions. Generally, it is similar to the idea behind temporal inactivity sessions – users will work on the same information need for an amount of time before moving on to the next information need – the difference being that the fixed temporal sessions assume a fixed period of time of working on the same need instead of trying to estimate the time that could potentially pass before the user starts working on a new information need.

For the use case at hand, the assumption is that a fixed total threshold will perform as well as the inactivity timeouts. The specifics of the data, especially regarding the frequency distributions of **user_ids** coming back to the website, suggest a comparable outcome with few differences regarding session length and session duration. Experiments were conducted using the same values as the inactivity timeouts. Since the 14 values tested previously are in a relatively wide range, this selection should also suffice as a total page-stay threshold. The concept is displayed again in Figure 4.5, where the maximum length is indicated by the frames. Here, the six interactions are grouped into three different sessions because the respective maximum length implies the structure.

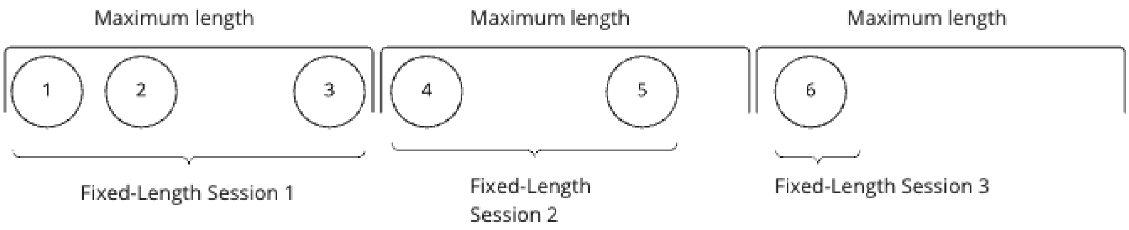


Figure 4.5: Example of the fixed-length session method. There are six interactions grouped into three sessions. The maximum-length frames indicate the length of each session, measured by the respective first interaction. Although interactions 3 and 4 are closer in time than interactions 3 and 1, they have not been grouped into one session. This is due to the maximum time expiring that begins from the point of interaction.

Again, the calculations were conducted using Spark SQL executed by a PySpark script on an EMR cluster. The code for implementing a fixed total threshold is similar to the implementation of the visit or path-based approach. Instead of comparing **url** and **http_referer**, the timestamps in the form of the field **tracetime** were compared between different subsequent rows of a **user_id** object.

For every fixed threshold, the timestamps between subsequent rows are compared. The first row of a **user_id** object is always the first reference **tracetime**. From then on, every **tracetime** that follows is compared to this first reference **tracetime**. If the difference between new **tracetime** and the old reference **tracetime** exceeds the threshold, a new **session_id** will be set. Additionally, the **tracetime** of the row will be the new reference **tracetime** to which all the following **tracetimes** will be compared. Following this, the first **tracetime** of every respective session is concatenated with the **user_id** to construct a unique id. This id is then assigned to every interaction of the respective session.

As an additional value, **session_days** are calculated as another variant of the fixed-time boundary. Session days refer to the actual date on which an interaction was performed. Practically, this equals the already-calculated **interaction_day** which is now combined with the **user_id** and assigned to every interaction on the respective day. Using the **interaction_day** as a session boundary introduces a hard boundary at midnight. Users browsing around midnight would then have interactions from two sessions. This approach is tested as it is most likely the simplest form of session identification. Simply, it measures the frequency of **user_ids** visiting the website in a commonly used time format, therefore making it easily computable and understandable.

4.4.2.4 Temporal – Dynamic Timeout

The next session types use a variable dynamic timeout (variants are identified with the prefix td, rows 29–37 in Table A2). Dynamic timeout thresholds are assumed to be more plausible than the fixed-inactivity timeouts since they may take the specifics of the dataset into account. For example, users of a newsletter website would probably spend more time on various pages than users of a weather website or an online encyclopedia such as Wikipedia. The assumption for these dynamic timeout sessions differs only in the sense that the temporal inactivity sessions' threshold may be dependent on certain conditions.

To obtain the different dynamic thresholds, a variety of control features was used to reflect the conditions that influence either a longer or shorter threshold.

The **page_template** was identified as the most important factor in controlling the timeout threshold. The various **page_templates** are representative of the website's business model and should therefore a) have clear differences in browsing behaviour on them and b) be weighted differently according to their importance regarding the business model. Another factor relates to the various product categories in the data to reflect the assumption that users spend different lengths of time browsing different categories. For example, a user may spend more time on pages when looking for computer parts compared to looking for a new gaming console. The difference when looking for a specific product or comparing different items should be visible in the data.

Therefore, multiple combinations of factors were used to calculate dynamic timeout thresholds. Per combination of factors, the average time spent on an interaction with the respective combination up to the next chronologically following interaction was calculated as the threshold for these factors and saved in a lookup table. The results were then joined to the original dataset and compared to the already calculated **timespan**. If the **timespan** exceeded the threshold value, a new session was set for the respective combination.

Since the **page_template** and the **category_id** information are arguably the most decisive factors in considering time spent on the website, both factors are used separately and in various combinations. In addition, a couple of other variants were considered: the time of year (where the day of the interaction was represented by the month) was chosen to potentially capture different behaviours relating to season; and the **device**, to be able to differentiate between mobile and desktop usage. The list below includes all combinations of factors to calculate average time:

- per **page_template**
- per **category_id**
- per **root_category_id**
- per **page_template** and **category_id**
- per **page_template** and **root_category_id**
- per **page_template** and month of **interaction_day**
- per **category_id** and month of **interaction_day**
- per **page_template** and **device** used for browsing (e.g. computer or mobile phone)
- per **page_template** and **category_id** and **device** used for browsing (e.g. computer or mobile phone)

Other approaches have attempted to come up with a dynamic threshold on a user basis [182]. Calculating a threshold per user might be reasonable when there are many interactions on a user level, i.e. having a lot of interactions per user on a specific set of pages.

A potentially workable example may lie in any streaming system – here, a lot of data per identifiable user would be available that could be used for better measuring the per-user thresholds. In the dataset at hand, the majority of users do not visit the system regularly.

4.4.3 Modelling Logical Sessions

This section details the steps that were taken to identify logical sessions. It explains the necessary preprocessing actions and describes the implementation of various mechanics to estimate topical similarity between different interactions. The methods differentiate between lexical and semantic session identification. In this dissertation, lexical similarity refers to the closeness of two or more (string) objects in terms of character overlap. Completely matching strings would indicate a lexical similarity of value 1, for example. Semantic similarity refers to the relatedness of two or more (string) objects regarding their meaning or context. Two semantically similar objects would have a similar meaning, for example, the terms smartphone and mobile phone.

4.4.3.1 Lexical and Semantic Properties in the Dataset

Most of the session-identification approaches presented in 2.3 deal with the identification of sessions in a search environment. The main focus is on users interacting with search engines to fulfil information needs. Logged interactions with a search engine are mostly queries, often, but not always, supplemented by any following clicks on results. Queries are the main component when trying to identify logical sessions (whereas a timestamp and a user identifier would suffice for temporal mechanical sessions). This is also reflected in the datasets, most of which originate from search engines such as AOL or Yahoo. In the case of the data used in this dissertation, queries are not the main component. Technically, a price comparison website could also be seen (and used) as a search engine. But practically, this is not – at least not in the main – the case here. Where other datasets can solely rely on queries, the dataset at hand is limited regarding this aspect of data. While queries are part of the data, the role they play is less important than the actual content of the visited pages. Of more direct value here are the fields that represent the content of the visited page. This includes all fields that contain names and identifiers related to the content of the visited page, such as the manufacturer, name or id of the product and especially the various fields related to the visited categories. Since the case in hand is a price comparison platform, queries are typically related to specific products or, more generally, to categories or manufacturers. While there will be more ambiguous queries for specific (or general) topics or areas such as, for example, 'star wars', nonetheless, the majority of interactions will be on product or category pages.

Measuring (topical) similarity between interactions for the dataset at hand is complex, since there is no real user input to compare to query logs that can easily be utilized. While queries do exist, they only make up 15.25% of all interactions. A much larger portion of the events consists of interactions related to the product page (41.13% *pageimpressions*, 18.07% to *leadouts*) and to a lesser extent to category pages (11.57%). Even the homepage gets a relatively high portion of the interactions with 6.82%. These percentages confirm that queries alone are unlikely to be able to detect coherent topical segments in the data. This contrasts with the strategy common to other published research that almost exclusively work with queries (i.e. [80, 88, 89, 120, 155]). For the dataset at hand, information relating to the content of the pages is far more relevant. This is why the approaches in this section identify sessions according to the semantic-similarity properties of visited pages instead of queries.

This rule refers to any identifier in the data. Since identifiers are available in nearly every trace after preprocessing, they can be used to compare the different interactions of a user. The same is true for the corresponding name fields, for example the **category_name** *Smartphones* (mobile phones) belonging to the **category_id** *19116*. The names may improve comparison between different levels of possible names since they have a semantic meaning. While there is no meaning to the id *19116*, the term smartphone can not only be found not only in the **category_name**, but also in the **product_name** and even in the **query** field.

With this information in mind, the construction of logical sessions with the data at hand becomes theoretically comparably easier than having to solely rely on **query** events, although further preprocessing is required. Using information from the category tree makes it easier to identify hierarchical connections between different events without necessarily having to interpret the meaning of a query. This is particularly useful for semantic connections; thus, whereas comparison between different queries requires a lot more context to identify similarity, the hierarchy of the category tree for the data at hand may already adequately demonstrate useful associations, as seen in Figure 4.6.

The structure in Figure 4.6 illustrates the utilization of the category tree. The different interactions such as *pageimpression*, *leadout* or *useraction* all come with an associated **category_id** defined by the product category (PC1, PC2, PC3 and PC4). Using the respective higher level per identifier (either subcategory (SC1, SC2, SC3) or root category (RC1, RC2)) in comparison to the hierarchy of the category tree leads to new possible connections using the associated information gained from the tree. The different events of the two visits are all associated with the different product categories and are, therefore, connected to the respective branches of the overall tree. In the example, the last two interactions of visit 1 could be connected logically to all already-connected interactions of visit 2, as they belong to PC2, which is connected to the rest of the interactions in this visit by originating in the same root category (RC2). The result would be two logical (lexical) sessions identified through lexical similarity by exploiting the matching **category_ids** as shown in the figure.

This constellation and the hierarchy have a clear advantage compared to using queries because all the events are already connected topically. Only the search result pages with the associated queries and the homepage need to be connected to the category tree since they have no tracked meaningful identifier. Any other **page_templates** that do not have an associated **category_id** (for example, because of erroneous tracking) should also be connected by either setting a placeholder id or extracting identifiers from the **url**.

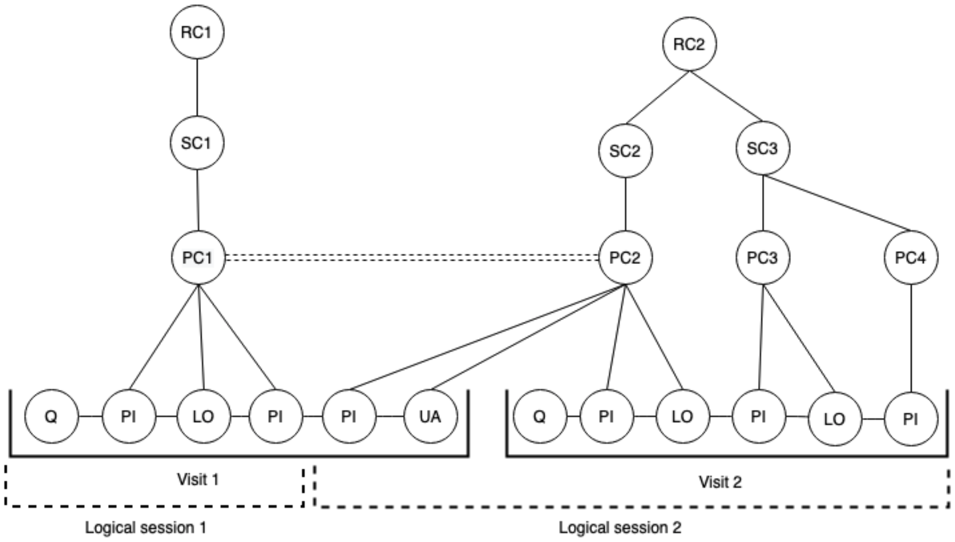


Figure 4.6: Example of topical connections using the category tree. The two dotted lines connecting PC1 and PC2 implies a potential connection between **category_ids** that are not in the same tree branch, whereas the fixed lines display a connection over direct **category_id** relationships among categories and with regards to their root category. Illustrated are two visits (connected via **url** and **http_referer**), that are actually two topically connected sessions because of the shared category branches. Abbreviations: Q = Query, PI = Pageimpression, LO = Leadout, PC = Product Category, SC = Subcategory, RC = Root Category, UA = Useraction.

Alternatively, or as an additional improvement to the quality of these already existing topical segments, different categories of the category tree could also be connected independently of the actual tree structure in the way queries also need to be somehow connected. This means, for example, related categories that reside in a different branch of the category tree. In Figure 4.6, this could mean that PC1 is actually related to PC2 as indicated by the dotted line between them – connecting all events to one big logical session. Understanding this connection effectively represents the construction of semantic similarity, which can then be used to construct even broader logical (semantic) sessions.

Finding the connection can be done via similar preprocessing steps to the identification of query similarity, resulting in two further preprocessing tasks that need to be completed before construction of actual logical sessions can begin:

1. Connecting queries to the category tree

2. Calculate similarity between categories independently from the category tree

The first task is important to avoid a non-trivial number of events with no assigned **category_id**. Although there are comparatively few query events, it is not unreasonable to assume that these may be a starting point for many longer sequences of events. In any case, connecting the queries to categories is an essential preparatory step to properly accomplishing the second task.

Both tasks are modelled as classic information-retrieval tasks. The first approach is based on the idea that finding the most relevant category for a given query is a very traditional information-retrieval task; having a set of terms and finding the most relevant documents essentially describes the task being undertaken here. The second task picks up on the idea that syntactically close terms are related or similar in natural language. Adapted to the setting here, the assumption is that categories interacted with subsequently are related or similar. For the first task, a variant of the original BM25 [223] retrieval algorithm was implemented via PySpark and SQL to gather matching **category_ids** to **queries** from a combination of different text corpora. For the second task, word2vec [186] as well as BM25 scoring were used to calculate the similarity between different categories. Introducing a novel idea, word2vec was applied on different category interaction sequences, effectively converting user traffic to vectors and utilizing them to estimate similarity.

It is important to note that this set-up deviates from using an explicitly formulated information need (i.e. a query). In this case, the information need of the user is extracted indirectly from information about user behaviour and navigation, implicitly assuming that the user is aware of what s/he is doing. Using the indirect information in the form of the **category_ids** assumes that the browsing user already knows to which category their information need belongs. This results in interactions that actually belong to the same information need being assigned to different sessions because the user did not explicitly know where their information need would be fulfilled. This is an interesting limitation that may result in multiple session breaks for logical sessions for some users. Considering the implemented method described in the following sections, it is assumed that this problem is at least somewhat mitigated by using an embedding algorithm. A further assumption is made about the systems employed to navigate to the information system in question: the information system itself will somewhat support the user in their behaviour (e.g. by routing even ambiguous queries in search engines to the 'correct' pages), thus minimizing the impact of the problem. It is notable that this impact could be analysed by implementing logical sessions. The following section describes the necessary steps to connect events without a meaningful **category_id** to the category tree, as well as how to calculate a category-similarity vector embedding.

4.4.3.2 Connecting Queries to the Category Tree

The first step in setting up an experimental information-retrieval task to find fitting **category_ids** for user queries is to create a suitable text corpus that can be used to match the said queries to associated **category_ids**. Theoretically, the **category_ids** could be extracted from the results the user clicks on after the query, but since many queries are not followed immediately by other events and the result sets are frequently very diverse, this was not the direction the current dissertation followed.

To construct the text corpus, several sources were used. The most obvious source is the dataset itself. Information about **category_names**, **category_synonyms**, **manufacturer_names**, **product_names** as well as **product_types** were extracted and stored with the associated **category_id**. In this case, the latter served as the document identifier when compared to a traditional retrieval system. Having collected this first batch of text,

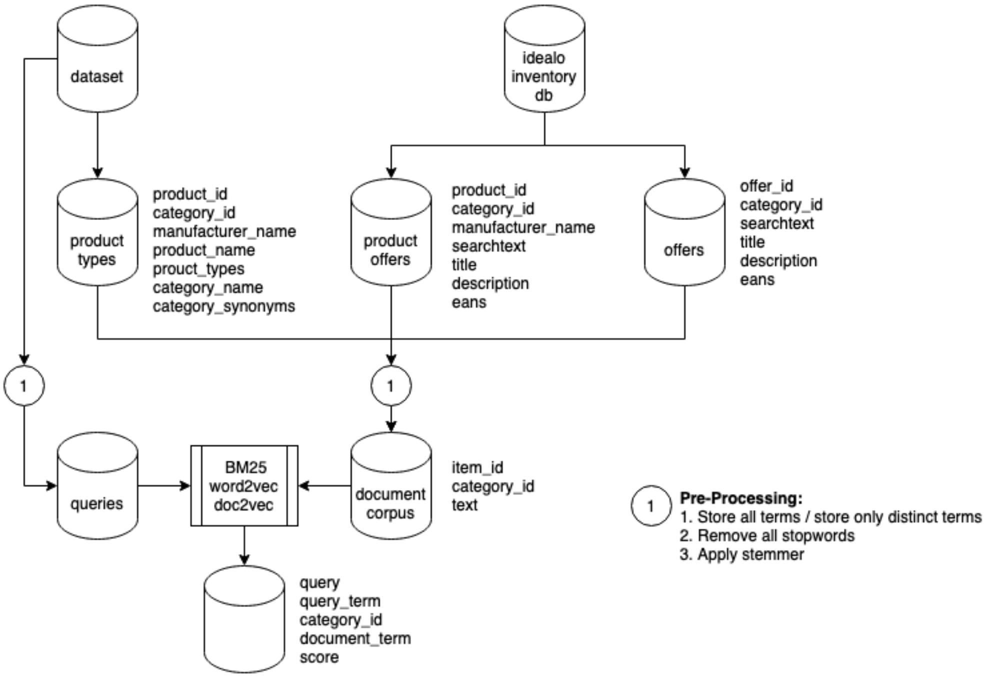


Figure 4.7: Preprocessing flow for text-data collection to match queries to the resulting document corpus.

the second step involved accessing the internal inventory databases from idealo to gather more data. This data was then separated into two buckets: one for the products and one for the offer data. This was necessary because although theoretically these datasets are identical since the product data is generated by the mapped offers, the internal structure is different. Both data sources were then used to extract further information. Figure 4.7 depicts the complete data and retrieval flow.

The number of extracted rows per **product_id**, **offer_id** and **category_id** was restricted to minimize the weight put on the frequently changing inventory; where a lot of price changes occur, evidence is often found in new rows in the inventory results for the same offer, product and end-category. For example, the category *Smartphones* has a much more volatile price structure than the category *Katzenfutter* (cat food), resulting in a lot more rows with the same information aside from the price.

From a content perspective, all fields containing meaningful text were extracted. This includes the following fields: **manufacturer_name** (only for the product objects), fields called **searchtext** and **description**, **title** and a separate field for **eans** (European Article Number or EAN). An example of a row from an offer object can be seen in Table 4.1. The **searchtext** and **title** are often, though not always, very similar and represent primarily a short description of the item, while the **description** field is a free-text field that describes the item at more length. The **eans** are collected in order to capture EAN queries. All these fields are then concatenated into a single **text** field and stored in two variants: one with all text included for every component; the other using only distinct terms – to affect

Field	Example
item_id	3333571109695003970
category_id	26849
searchtext	Airfryer – Die besten Rezepte Anne Peters
description	Viel Genuss mit wenig Fett . Gesund frittieren, abwechslungsreich genießen – die besten Ideen für die Heißluftfritteuse . Tolle Snacks, Gerichte und Desserts für das innovative Küchengerät . Über 35 Rezepte von Pommes bis Popcorn, von Chicken Wings bis Backofen-Tortilla, von herzhaft bis süß . Mit zahlreichen brillanten Fotos Viel mehr als nur heiße Luft: Der Airfryer ist ideal für alle, die sich abwechslungsreich und gesundheitsbewusst ernähren möchten. Statt in Öl garen die Zutaten in einem heißen Luftstrom zu krossen oder saftigen Köstlichkeiten mit natürlichem Aroma. Und sparsamer, als für kleinere Portionen den Backofen anzuwerfen, ist dieses geniale Küchengerät allemal. In diesem Buch finden Sie tolle Ideen für das beliebte Multitalent. Ganz ohne oder mit nur wenig Fett gelingen Pommes, Chicken und Gemüsechips auf Knopfdruck. Auch süße Leckereien wie Brownies, Bratäpfel oder Popcorn lassen sich in der Heißluftfritteuse perfekt zubereiten. Entdecken Sie die neuen und extrem vielfältigen Möglichkeiten für puren Genuss!
title	Airfryer – Die besten Rezepte Anne Peters ePUB
ean	9783815554234

Table 4.1: Example of an offer item from the document corpus.

the frequency of the term associated with certain terms – for the offer data as well as for the product data.

The preprocessing conducted while saving the text is depicted in Figure 4.7 as a number 1 inside a circle. In this stage, the number of characters was reduced to no more than 100, redundant white spaces were stripped and all special characters were removed. Stemming, stopword removal and a combination of both was also applied on the different corpus variants. As for the stopwords, a generic German stopword[18] list was used, complemented by a small number of e-commerce-specific terms like *preisvergleich* (price comparison), *kaufen* (buy it) or *günstig* (cheap), but these are not relevant to the task at hand. For stemming, the Cistem stemmer [271] from the Natural Language Toolkit (NLTK) Python library [19] was used. Another step included aggregating all text to **category_id** level, thereby drastically reducing the number of documents in the corpus, since now every document represents a **category_id**. After preprocessing, 16 different text configurations (i.e. different combinations of preprocessing) were used, on which to to run the retrieval model.

Now the BM25 retrieval algorithm was implemented using Pyspark and SQL to match the queries from the dataset with the document corpus. PySpark was used instead of existing libraries because of the quantity of data that needed to be processed. Several different formulas were tested following the insights from Kamphuis et al. [122], thereby also testing different calculations for the inverse document frequency. The resulting scores per term were then used in combination with all query terms to calculate a final score per

[18] `https://solariz.de`, retrieved 10 December 2020. No longer accessible.

[19] `https://www.nltk.org/_modules/nltk/stem/cistem.html`, retrieved 10 December 2020.

respective **document_id**. Testing was carried out on the original BM25 formula, simple tf/idf scoring as well as BM25L [161] on the supposition that the latter would normalize the penalty on longer documents, using the following formulas:

- BM25
$$\sum_{t \in q} \log\left(1 + \frac{N - df_t + 0.5}{df_t + 0.5}\right) \cdot \frac{tf_{td}}{k_1 \cdot \left(1 - b + b \cdot \left(\frac{L_d}{L_{avg}}\right)\right) + tf_{td}}$$
- BM25L
$$\sum_{t \in q} \log\left(\frac{N + 1}{df_t + 0.5}\right) \cdot \frac{(k_1 + 1) \cdot (c_{td} + \delta)}{k_1 + c_{td} + \delta}$$
- TF*IDF
$$\sum_{t \in q} TF_t \cdot IDF_t$$

Where df_t is the number of documents d containing a term t, N is the number of all documents (i.e. the number of **category_ids**), tf_{td} is the term frequency of term t in document d, L_d is the number of tokens in document d, L_{avg} is the average number of tokens in relation to the complete corpus, c_{td} a reformulated tf_{td} component in BM25L and k_1, b, δ are free parameters. c_{td} equals $tf_{td}/(1 - b + b \cdot (L_d/L_{avg}))$. k_1 acts as a smoothing parameter for term-frequency weights whereas b is used as a normalizer for document lengths. Similarly, δ is supposed to give longer documents a boost.

Since query-matching quality is technically not within the scope of this dissertation, a complete evaluation of the retrieval algorithm and the ranking results was not performed. Had this been the case, a full evaluation would have involved setting up the appropriate test collection using manual relevance-assessment for a set of representative queries. Since the list of queries contains 45,416,965 distinct queries, with 7,675,830 distinct terms and 147,295,562 terms overall, setting up a valid evaluation environment would have been a project in and by itself. Instead, for simplicity, 20 specifically chosen test queries were used to determine if the results of the scoring were valid enough. The queries were checked against the first 20 retrieved documents (i.e. categories).

After testing the respective corpora and different scores, the best results were achieved subjectively on the corpus by retaining all terms from the original data flow, with stopwords removed and aggregated at a **category_id** level. As for the retrieval score, BM25L seemed to deliver the best results with a strong normalizer on document length, which is reasonable considering the structure of the corpus. Documents representing categories that have a volatile price structure included many more terms in much higher frequencies compared to those with fewer price-changing categories. To normalize the impact of the document length, the respective parameter of BM25L seemed to do the trick here. Parameter b was set to 0.00001 and k_1 was set to 50 after multiple experiments, in an attempt to estimate their impact on the ranking outcome. δ was left at the standard value 0.5. Nonetheless, another addition was made after reviewing the results and to reflect the importance of the categories according to the dataset and the website itself. A fixed parameter was added to the score weighting to account for the flow of traffic a respective category received. This traffic normalizer was calculated by counting the interactions per **category_id** in the dataset and calculating the z-score[20] of the result.

The results lead to some rather interesting implications. Apparently, the document corpus has some very specific properties. This is only logical since it basically is a concatenation of three different text objects. The massive differences in document length introduce a certain error margin as well. One assumption is that such a corpus needs many more e-commerce-specific stopwords to reduce the noise in order to deliver stable results. Another assumption is that the text itself is not necessarily very descriptive of the categories. Often there are some strange connections between different categories.

Since this task is only the first of two preprocessing tasks, the results were deemed satisfying enough – however, in a productive setting, these should be tuned and tested further and, even more importantly, be carefully evaluated. Having logical sessions identified in a productive environment, the information gained could be used to connect incoming user queries to the existing categories even more precisely. In the task at hand, the results from the BM25L score were identified as having adequate quality.

As the last step, the topmost similar **category_id** per query was joined back to the dataset. For around 3,282,939 queries, no **category_id** could be identified, resulting in around 0.001% events without a meaningful **category_id**. Every event missing a **category_id** is replaced with the value *42* in the dataset. The results can then be used for the second task, which is explained now.

4.4.3.3 Calculating Similarity Between Different Categories Independently from the Category Tree

This second task is a major step towards identifying logical sessions. The first task is (relatively speaking) merely a preprocessing step that should normally already have been established in the data since the underlying retrieval algorithm (for user queries) ought already to have the necessary information from the productive system. However, the second task is much more important because this is where the topical connection between events is defined. Multiple approaches were tested.

The first experiment is a novel approach introduced for the first time in this dissertation to the author's best knowledge, very similar to embeddings used in recommendation applications (search-context embeddings for ad query matching [84] or product-user embeddings for product recommendation [85]). By introducing a similarity system of associated categories, topical connections between categories (and queries) are easily identifiable. The presented approach utilizes the already-existing traffic to estimate category similarity in an unsupervised way: the idea is to base the similarity association in the context of a **category_id**'s surrounding categories on user behaviour. By doing so, it is possible to identify category relations that share a common space in relation to the information needs of the user. In fact, the assumption here is similar to the base assumption of mechanical sessions using time boundaries: users will work on information needs in small sequences of

[20] `https://en.wikipedia.org/wiki/Standard_score`, retrieved 28 November 2021.

events and, therefore, will frequently visit related categories in a sequential manner before moving on to other categories. An example could be buying a new smartphone and looking for associated accessories. Utilizing the frequency of all users to identify such relationships introduces an elegant way of identifying topical segments without having to rely on an arbitrarily chosen amount of time with no knowledge of the information need.

A second experiment utilizes the shared-term space of all **category_ids** in order to compute their similarity. This is a more traditional approach, comparable to concepts in the literature used to calculate query similarity (for example, as presented by Gabrilovich and Markovitch [79] where the authors call a comparable approach Explicit Semantic Analysis to augment text representations with knowledge extracted from Wikipedia). The BM25L scoring for the document corpus from task one is used again to calculate the most relevant **document_ids** and, therefore, similar **category_ids** for all terms from the **category_names**. Additionally, the top 10 scoring words for every **category_id** were used as well to retrieve similar categories for highly specific **category_names**[21]. This led to a similarity ranking of **category_ids** based on shared terms. The same could have been done via word2vec or doc2vec, but this step was skipped here since the results from the BM25 algorithm were already good enough; however, in a productive system, this may be a starting point for further tuning and optimization.

user_id	user sequences
105151986	26575, 100, 9152, 16455, 10192, 25622, 3309, 10192, 25357, 25375, 1840, 31231, 9792, 13172, 26671, 19116, 23278, 14232
103092731	15701, 4012, 15701, 4012, 15701
103105399	19116, 9552, 19116, 9552, 2925, 30890, 18677, 6073, 9552, 1, 9552, 25843, 25739, 3972, 14673, 8912, 32497, 25962

Table 4.2: Example of user category_id sequences.

The data context consists of the complete interaction sequences of users. The problem with these sequences may lie in the time distance between each interaction, which eventually could potentially lead to unrelated categories. On the contrary, this might have no great effect because all the traffic of all users is used. Furthermore, the impact might be lessened assuming that users will visit different categories after a break. A temporal correction factor could solve this problem; for now, the data is considered to be meaningful without any corrections. An example of a sequence can be seen in Table 4.2.

In the example, **category_ids** are aggregated per respective identifier in order of their appearance in time. Consecutive repeated values are removed so that only one appearance is in the data. Interactions with no **category_id** as well as interactions with a **category_id** with no inherent meaning[22] are excluded. As another approach, **manufacturer_ids** or even **product_ids** could be used instead of **category_ids**, potentially also in combination

[21] Such as for the category *Diabetikerzubehör* (diabetic accessories), where no frequent offer updates and low traffic leads to no appearances of the term in the corpus.

[22] Meaning all **category_ids** below 300, as these represent the system pages like the homepage, privacy policy or the imprint, or have a placeholder **category_id**.

with them when the lower level identifier is missing. Since using the **category_id** is seen as a valid enough approach to define the topical connection between different events (and therefore information needs), experiments with these identifiers were omitted. Still, these could be used in order to create a far more fine-grained map of identifiers, leading to more closely separated segments. In the e-commerce setting at hand, **category_ids** should be good enough.

Having already readied the necessary data, word2vec [186] using the Gensim [219] implementation was applied on the sequences. A first experiment was conducted employing the continuous bag of words (CBOW) learning algorithm, using vector sizes of 70, 100 and 300 and window sizes of 2, 3, 5, 10 and 1,000, in 10 iterations. The second run of experiments was conducted with the Skip-gram learning algorithm, using vector sizes of 70 and 100 and window sizes of 2, 3, 10 and 1,000 and negative sampling of 20, in five iterations.

Although no extensive parameter tuning was conducted, the tested parameters were chosen based on some assumptions. Following the models described by Mikolov et al. [186], both presented architectures were tested under the hypothesis that Skip-gram will deliver more fitting results than CBOW. Skip-gram architectures are intended to predict surrounding words given a reference word, while the CBOW architecture predicts a reference word based on its context. With this in mind, a Skip-gram architecture is likely to be more suitable for the task at hand. The different model parameters were decided based on the size of the vocabulary and the findings from Levy and Goldberg [135], where the authors argue that larger window sizes (i.e. of 5 and above) capture broader topical content in comparison to smaller sizes, the latter being more appropriate for gathering specific information regarding the target word. This said, other works do not observe notable changes of similarity between different window sizes [72].

For the application at hand, it was decided that, theoretically at least, the bigger window sizes would deliver better results, hence the tested value range was chosen generously with the window size 5 being the standard in the Gensim library. In terms of the vector size, there seems to be evidence that dimensionality does have an impact on the embeddings in relation to the vocabulary size; Patel and Bhattacharyya [203], for example, argue that the performance of the embedding will be affected until a lower bound is reached defined by the vocabulary size, after which the performance stabilizes. Generally, a dimensionality of 300 seems to be the standard in the literature for corpora with a much bigger sized vocabulary; because the size of the vocabulary here is only about 2,300, the three values tested were deemed adequate. The assumption being that the difference should be minimal due to the small-size vocabulary.

Since these embedding algorithms are unsupervised and present no real possibility for evaluation aside from applying them on downstream applications and evaluating those, a similar evaluation as in task one was conducted. The results from a sample of 100 randomly chosen **category_ids** were manually examined to check their integrity. Devising a more formalized way of doing this would have been preferable, but was beyond the scope of this dissertation. The same is true for the other variant of achieving similar categories using the BM25 scoring. Here, all terms of the **category_name** are used to get a relevance scoring from the document corpus on **category_id** level. The top 25 most similar categories according to the BM25L ranking were then extracted for the 100 randomly chosen categories and manually checked. After checking the results, the similarities derived from the model based on the CBOW architecture with a vector length of 300, a window size of 10, in 10 iterations and no other parameters, were deemed sufficient enough.

The final step before the similarity associations are applied in the logical session-identification approaches is to define a proper similarity threshold. The question as to what threshold should be set to decide whether two categories are similar enough to belong to the same session is a rather difficult one to answer, since it is completely dependent on the underlying data basis and usually cannot be generalized. The similarity score between different terms of the vocabulary in Gensim is based on the cosine similarity ($\cos \varphi$) between the generated vectors. Generally, the vocabulary cardinality and the used dataset as well as the parameters have a strong effect on the similarity results [40]. The differences in similarity values between two **category_ids** c_1 and c_2 are not easily understandable.

category_name	cosine similarity
Handy	0.76
Telekommunikation	0.73
Handy-Ersatzteil	0.70
Displayschutzfolie	0.70
Handytaschen	0.70
Kabelloses Ladegerät	0.66
Tablet	0.56
Sonstiges Handyzubehör	0.55
Handy-Akku	0.52
Handy-Fahrradhalterung	0.51

Table 4.3: Example of the top 10 most similar categories of category_id 19116 (Smartphones) according to embeddings based on the complete history of users.

The problem is illustrated in Table 4.3. As can be seen there, the increments of the cosine similarity between the different, supposedly similar categories do not seem to follow an easily understandable pattern. At least at first glance, there is no reason why *Handytaschen* (mobile phone cases) would be more similar than, for example, *Handy-Akku* (mobile phone batteries). The cosine similarity is not easily interpretable and, more importantly, not easily evaluated. Especially when considering that the calculated similarity is actually defined by user traffic – similarity here is subjective, as it reflects the relationship between categories according to user behaviour. Although there seems to be a meaningful pattern behind these increments according to Elekes et al. [72] (at least in natural language contexts), the score does not reflect any easily detectable relationship based on similarity.

This could be a point where tuning could be implemented to optimize the actual algorithm, since there seems to be a lot of potential here. There again, training another statistical model (i.e. logistic regression or an RNN) might be an option here to identify a general threshold per **category_id**. Another approach would be just to take every association above a certain value. While this may be another valid option, it is kind of arbitrary nonetheless since the values are dependent on the model and its parameters as well as on the vocabulary. For the use case at hand, different variants of a threshold point are experimented with, as shown in the following section.

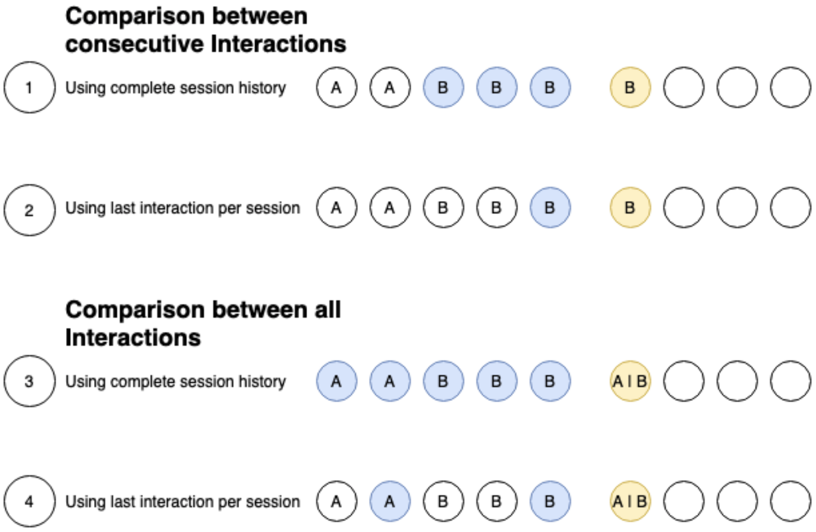

Comparison between consecutive Interactions
1
Using complete session history
A
A
B
B
B
B
2
Using last interaction per session
A
A
B
B
B
B
Comparison between all Interactions
3
Using complete session history
A
A
B
B
B
A | B
4
Using last interaction per session
A
A
B
B
B
A | B

Figure 4.8: Example of the logical session-identification comparison contexts. A and B represent different sessions. The blue circles represent the respective comparison context. The yellow circles represent the reference event that is compared to the blue circles. The white circles are future events that will be compared afterwards.

4.4.3.4 Identifying Logical Sessions

The calculated similarity can be utilized in various ways. This dissertation employs a variety of different mechanics that reflect the different variants of potential topical connections. Following the basic premise that users may work on information needs in a stop-and-go manner, this research differentiates between consecutive and interleaving sessions for logical sessions. Consecutive sessions consist, as the name implies, only of consecutive interactions. If the topical connection between events cannot be made, the respective session ends and any new interactions interacting with the same **category_ids** become a new session. Interleaving sessions, on the other hand, employ the concept of the user working on an information need, then working on a different need and returning to work on the original one later.

Additionally, one can differentiate between simple (logical) sessions that deal specifically with one particular topic area and the more complex journeys that encase various related information needs. This dissertation implements the differences between these two concepts in the (simplified) form of a 'comparison context'. Comparison context refers to the comparison base between different interactions (either consecutive or interleaving): direct comparison means that only the latest interaction of a session is compared to the next reference interaction; the complete history refers to using all past interactions of a session while comparing them to a reference interaction. This leads to the four different comparison concepts for logical sessions that are illustrated by Figure 4.8.

The different mechanics are intuitively understandable. Using the complete session history in consecutive comparisons as illustrated by mechanic 1 leads to self-contained topical units. These units represent coherent interactions belonging to the same topic defined by the complete session content; depending on the actual comparison between the two different components, this may lead to broad or very specific sessions. The second concept in Figure 4.8, mechanic 2, is stricter; comparing only the last interaction of the previous session with the following reference interaction may lead to very specific logical sessions. The idea is one of developing information needs so that only the last interaction defines the current sequence of interactions and task. Both context mechanics 1 and 2 are limited to the session preceding the reference event.

The same assumptions are true for mechanics 3 and 4 except that the resulting sessions may consist of interleaving branches instead of consecutive units, leading to even broader sessions and, eventually, in the case of concept 3, to complex journeys consisting of many different, somehow related information needs. Comparison context 4 is a special case: when comparing only the last interaction, the resulting logical session may represent an evolving journey. Comparing the general idea behind mechanics 3 and 4, 3 may result in endlessly continuing sessions whereas 4 may result in developments that are too specific, which neglect backtracking to specific parts of a journey. Either comparison method may be very prone to errors due to their identifying sessions too broadly or too narrowly. Both 3 and 4 use all sessions that took place before a reference event for comparison, so that the reference event could be connected to any of those or initiate a new session.

The comparison method between different interactions introduces the third dimension affecting the identification of logical sessions. This dimension, which eventually defines the nature of the resulting sessions, has an impact on the outcome as it identifies, either broadly or narrowly, very strictly defined or topically evolving sessions. Using the results of the previous sections, multiple comparison methods with different variants were selected to conduct several experiments:

- Baseline: lexical-matching utilizing **root_category_ids** (variants are identified by the prefix l, rows 38–39 in Table A2)
- BM25L shared-term space: top 10 similar categories according to relevance ranking (variants are identified by the prefix bm25, rows 40–43 in Table A2)
- **category_id** embeddings on user history (u2v): $\cos\varphi$: top 10 most similar categories (variants are identified by the prefix u2v10, rows 44–47 in Table A2)
- **category_id** embeddings on user history (u2v): $\cos\varphi > 0.5$ (variants are identified by the prefix u2v05, rows 48–51 in Table A2)
- **category_id** embeddings on user history (u2v):$\cos\varphi >$ cut-off parameter (variants are identified by the prefix u2vc, rows 52–55 in Table A2)

These comparison-method variants were selected to offer a broad experimental range comparable to mechanical approaches. Essentially, the baseline approach represents lexical sessions – only when the **category_ids** or **root_category_ids** of $interaction_i$ and

$interaction_{i+1}$ are matching[23], is a session identified, otherwise a new **session_id** is set. The second method uses the top 10 most similar categories according to the BM25L relevance ranking. The reason for the top-10 selection is based on the concept of information-retrieval relevance, whereby the higher the relevance score, the more important the topic is to another topic. In this case, **category_ids** that rank higher are assumed to be closer in terms of their vocabulary and therefore are assumed to be more similar.

The same assumption holds true for the other comparison methods. Taking the top 10 most similar categories according to the calculated $\cos\varphi$ is more or less an arbitrary value that again resembles the displayed results of a search engine. Using all similar categories above a $\cos\varphi$ of 0.5 is an approach commonly applied in the literature [83] dealing with any kind of word embedding; it is important to note that this value is usually applied to data with a natural language context, so it might not be a fitting option here. The cut-off parameter is a dynamically calculated threshold based on the distribution of the $\cos\varphi$ values per **category_id**. It was empirically defined while evaluating the results for the 100 test categories with the following formula:

$$p = max(\cos\varphi_{category\_id})/2 + 0.75 * \sigma(\cos\varphi_{category\_id})$$

This dynamically calculated formula was deemed to identify similar categories relatively accurately with respect to the respective nature of the **category_id** according to an empirical visual evaluation. For example, the category *Smartphones* (mobile phones) would have less similar categories than the category *PC-Komponenten* (computer parts). This makes sense as one is a product category with a of lot less-related categories whereas the other is a subcategory with potentially many more similar categories.

The combinations of all comparison contexts and comparison methods are reflected in the tested approaches as far as practicable. Every variant has been implemented reusing the same methodology that was employed for identifying visits and fixed temporal sessions. Instead of comparing **url** and **http_referer** or **tracetimes** from different rows of a **user_id**, now the **category_ids** of different rows are compared, according to their potential similarity with the respective mechanic.

Interactions with missing or placeholder **category_ids** (i.e. the homepage) are treated as special cases; events like this will be connected depending on the current session behaviour. Generally, all interactions with a placeholder are connected to the session that is closer in time. This might introduce errors, as it cannot be determined to which information need the interaction refers, but was deemed to be reasonable under the circumstances. Another exception is made for events that are more than 1,440 minutes (one day) away from other interactions with a meaningful **category_id**; since it cannot be determined to which (logical) session and therefore information need they belong, such interactions are identified as a separate session. The temporal boundary here was arbitrarily chosen;

[23] This is why only comparison contexts 2 and 4 were applied to the baseline: using the complete session history to match categories makes no difference here.

the assumption is that a full day between the reference event and another event with a meaningful **category_id** is a valid gap to assume a different information need.

Theoretically and under ideal circumstances, such logical sessions could be clearly defined with a precise endpoint; for example, with a purchasing event. However, considering the quality of the dataset and the nature of the price-comparison business, this issue is not broached for the time being, as it is not entirely clear how users interact with the website. Using logical sessions in the implementation as explained, therefore, potentially could create never-ending or overly long sessions, but these could be bypassed with clearly defined endings.

Multiple variants of session-identification approaches have been tested. Sessions identified by lexical similarity are technically mechanically divided segments but they do have a logical component because of their basic assumption. A lexical connection between subsequent events indicates a topical connection because the connection assumes that the events are related to the same information need. According to Figure 4.1, lexical sessions could be a fragment of a bigger logical session or a journey, but they could also simply represent a self-contained logical session – all depending on the definition. The baseline method represents lexical approaches whereas the other methods are semantic by their nature.

This sums up to two baselines representing lexical sessions, four variants utilizing the shared-term space according to BM25L ranking and 12 approaches employing the $\cos\varphi$ from the user category embeddings. Overall, 18 different purely logical approaches have been tested. All approaches can be found in Table A2, marked with an L (for logical sessions).

4.4.4 Modelling Combined Approaches

In this section, some variants of session identification that were presented in Section 2.3 are tested on the dataset alongside an expanded variety of approaches based on the logical session-identification methods in the previous section. As explained in the previous section, for most of common logical session-identification approaches, due to the lack of queries, it is hard to transfer the actual concept. Where the majority of approaches relies on queries to separate logs, this is not a viable strategy here. Nevertheless, as far as it is possible the following section will attempt to transfer the key points of a small subset of algorithms.

A potential approach in the literature was presented by Mehrotra et al. [178] and Halfaker et al. [91]. The authors propose a Gaussian Mixture model to identify a general temporal threshold for the user basis. Mehrotra et al. [178] work with data coming from interactions with digital assistants like Siri[24] or Cortana[25] whereas Halfaker et al. [91] test their method on a variety of fields. The actual algorithm is based on inter-activity times per user, fitting a (two or more component) Gaussian Mixture Model on the scaled time spans. A good threshold would then be where the inter-activity time is equally likely to be in either

[24] `https://www.apple.com/uk/siri/`, retrieved 28 November 2021.

[25] `https://support.microsoft.com/en-us/topic/what-is-cortana-953e648d-5668-e017-1341-7f26f7d0f825`, retrieved 28 November 2021.

of the two components. Generally, this method is simple to apply. However, because the assumption of the current research is to find a general (user) threshold and the temporal thresholds tested already showed this approach offers no particular advantage over the predefined thresholds, this dissertation takes exploration of this method no further.

Gayo-Avello [80] offer another approach. The original variant is a combination of lexical similarity between query strings with a degrading temporal effect. The so-called geometric method calculates the temporal distance between query q_i and query q_{i+1} using the formula:

$$T_{distance} = \frac{t_{i+1} - t_i}{time\_limit}$$

Where t_i represents the timestamp of a query q_i and *time_limit* is a user defined threshold, which is set to 24 hours in the experiments. For computing lexical distance, the author treats queries as bags of character n-grams instead of bags of individual words. The algorithm compares the n-gram representations of a query q_{i+1} with all n-gram representations of the current session, calculating a ratio between zero (for no shared n-grams) and one if all n-grams from query q_{i+1} are already present in the session n-gram representation. The result is geometrically interpretable; between every query q_{i+1} and query q_i or session s_qi respectively the method calculates the values for temporal and lexical distance, resulting in two values in the ranges [0, 1] and [1, 0], defining a point in a 2D-vector space. Originating at point [1, 1], a unit circle is then drawn on both positive semi axes; all areas enclosed by this circle are then defined as a session continuation.

Using the same method [80] as a step in their own approach, Hagen et al. [88, 89] define query- and search-session detection in a cascading way. Their approach applies different steps one after another to decide on session boundaries. As a reminder, the following steps are applied in the refined and updated version from 2013 [88]:

1 Time-limited segments (using a 90-minute inactivity timeout)

2 Simple pattern comparison

3 Lexical similarity with a variant of the geometric method from [80]

4 Semantic similarity using ESA [79]

5 Semantic similarity using LOD

6 Comparing search results for queries

All these steps are applied to a search log to identify topical segments in a cascading way, deciding with a level of confidence upon query connection or separation before moving on to the next step. Afterwards, the authors apply them again on the now constructed logical sessions to find search missions. In terms of the data, Hagen et al. [88] only compare successive queries and topical segments for performance reasons; in a live online environment run time would be exceeded.

The difficulty once again is the lack of consistent query usage in the dataset at hand in that it restricts the application of, for example, the simple pattern step, which then makes later steps not directly applicable for the same reason. In essence, the cascading concept's role in this dissertation is minor because its aim predominantly is to improve performance of the algorithm and not necessarily to enhance the quality of the outcome. The objective of the cascade is to decide on query connection before moving on to the next step, which undoubtedly would involve more work and time. Hence, since algorithm performance is not within the scope of this dissertation, only a limited number of the later steps have been implemented. All concepts up to step five were already applied in previous sections in one way or the other.

The work of Jones and Klinkner [120] on session identification is one of the most influential in the area. They were among the first to introduce new terminology with terms like search session, search goal and search mission. Their approach aims to find interleaving, hierarchical constructs made up of goals and missions. In their annotated dataset, they employ a logistic regression model with a variety of different features as a binary classifier. Since the dataset at hand is not annotated, training a supervised model was not possible.

Unfortunately, most of the approaches discussed so far were not applicable, either because they overly rely on query data or, worse, rely on annotated training data to fit their models. As has been shown, however, the reliance on query data can be overcome in part by transferring the model methodology from queries to the category information. Most of the concepts presented to identify logical sessions are already inherently present in the approaches presented in Section 4.4.3. As for the algorithms relying on annotated data for training, these are more problematic for the reasons already discussed in Section 2.4. Since objective annotation of the dataset without implying certain topical segmentations is simply not possible, all of these approaches are left aside in this dissertation. Instead, this research adopts the similarity functions from the previous section and reuses an adapted version of the geometric method from Gayo-Avello [80]. This dissertation slightly extends the original method that calculated lexical distance by comparing the n-grams between successive queries and temporal distance in a normalized form as a degrading value from a maximum temporal threshold of 24 hours. Here, the temporal-distance calculation employs the same method, but uses different maximums: 24 hours, 14 days and 75 days. The 24 hours is used to replicate the original method. The 75 days represent the average time between *Direktkauf* orders calculated from the dataset. The 14 days are an arbitrarily chosen value. This dissertation, in approaching the lexical distance differently, reuses the $\cos\varphi$ between the **category_ids** of all interactions as a substitute for the lexical distance (instead of computing lexical similarity between query interaction n-grams). In another variance to the original method, the form of comparison contexts in use is also different. The original method compares the complete session history of a query q_i with a reference query q_{i+n}. This dissertation experiments with all combinations of the explained context concepts.

The actual geometric function is modelled after the interpretation of Hagen et al. [89], simply using interaction similarity instead of consecutive query similarity. Therefore, interaction i is similar to interaction j, if the condition of the following formula is met:

$$\sqrt{(f_{similarity})^2 + (f_{temporal})^2} \geq 1$$

This formula is a direct adoption of the original geometric method: if the sum of both distance functions is within the area enclosed by a unit circle in point [1, 1] in a geometric space, the algorithm continues the session. Otherwise, a new session is generated. This leads to the following approaches using a variant of the geometric method [80]:

- u2v cosine similarity between consecutive interactions (24 hours, 14 days) (variants are identified with the prefix and suffix geom + cc/cd, rows 56–59 in Table A2)
- u2v cosine similarity between all interactions (24 hours, 14 days, 75 days) (variants are identified with the prefix and suffix geom + ac/ad, rows 59–65 in Table A2)

Due to the popularity and widespread application of simple temporal thresholds between interactions as a means of session separation, these are tested as well using the approaches employed previously. The combination of logical sessions with a mechanical timeout may be a valid strategy to amalgamate the assumed advantages of both sides: that is, achieving the internal coherence of logical sessions without encountering the potential problem of never-ending sessions. Finding the best temporal threshold is dependent on the nature of the session approach. Applying a 30-minute timeout on logical sessions with interleaving interactions or even journeys combining multiple information needs (which are effectively represented by the embedding approaches) would not make that much sense, as the interleaving behaviour would be restricted to a very short time frame. Therefore, a selection of already-tested timeouts from the mechanical sessions in Section 4.4.2.2 are applied, but only to approaches comparing consecutive interactions. For the approaches that consider all interactions of a user, larger temporal thresholds are a more reasonable choice. Combinations of all following dimensions are tested:

- BM25L similarity between consecutive interactions (5, 30, 1,440 minutes and 14 days) (variants are identified with the prefix and suffix bm25 + ti + cc/cd, rows 122–127 in Table A2)
- u2v cosine similarity between consecutive interactions (5, 30, 1,440 minutes and 14 days) (variants are identified with the prefix and suffix u2v + ti + cc/cd, rows 74–97 in Table A2)
- Lexical matching between all/consecutive interactions (using **root_category_id**) (24 hours, 14 days, 75 days, 180 days) (variants are identified with the prefix and suffix lti + cd/ad, rows 66–73 in Table A2)

- BM25L similarity between all interactions (24 hours, 14 days, 75 days, 180 days) (variants are identified with the prefix and suffix bm25 + ti + ac/ad, rows 128–135 in Table A2)
- u2v cosine similarity between all interactions (24 hours, 14 days, 75 days, 180 days) (variants are identified with the prefix and suffix u2v + ti + ac/ad, rows 98–121 in Table A2)

The implementation is identical to the logical approaches: all **category_ids** of the user object are compared according to the accepted mechanic, base of comparison and, in addition, the period of time between interactions. Interactions with no **category_ids** or placeholder **category_ids** are again treated as special cases. Instead of the 1,440-minute temporal threshold used before, the time threshold used in the respective variant is employed; e.g. instead of assigning a new **session_id** to interactions with a placeholder **category_id** with a 1,440-minute temporal distance to an interaction with a meaningful event, this distance must be 14 days or even 180 days.

This concludes the section for the combined approaches. Overall, 80 different variants were implemented using the listed mechanics and the different comparison contexts. Now that all the **session_ids** for all the different approaches are in one place, they are ready to be analysed. The evaluation strategy is presented in the following section.

4.5 Evaluation

This section describes the steps that were taken to evaluate the differences between all session-identification approaches. All in all, 135 different variants were implemented. A complete overview of all approaches can be found in Table A2. First, some preliminary points regarding the evaluation of session approaches are discussed. Afterwards, the actual evaluation is described, which consists of two separate parts. The first part is a comparative analysis of all variants using specified measures. The second part consists of the application of multiple variants in an actual use case that may be relevant in a productive environment.

4.5.1 General Evaluation Approach

As presented in Section 2.4, the evaluation of session-identification algorithms is usually performed by comparing the resulting sessions with a previously annotated dataset. The comparison is carried out using typical measures such as precision or recall to assess correctly identified session continuations or stops respectively. This approach comes with a few flaws that need to be discussed here.

Using a gold-standard dataset to evaluate multiple session-detection approaches has several biases. One flaw to consider is the difference in the underlying base assumptions for every single identification approach that is compared to the gold standard. Necessarily, all the various approaches and mechanics have made different assumptions about user behaviour, which will probably impact the identified sessions. Comparing all these sessions to the same gold-standard sessions would likely lead to misinterpretations. The same assertion is valid for the gold standard itself – with which assumption in mind should the data be annotated? Ultimately, this leads to comparing sessions that cannot or should not be compared like this because the comparison method itself is biased. To be more precise, there is no right or wrong here; a gold standard implies the correctness of the annotated sessions, whereas the different session-identification approaches are all correct according to their underlying assumptions. The differences between them only become visible by measuring their impact on certain measures or use cases, that is, here, the attempt to understand user behaviour.

Another point to consider is the type of evaluation and the hypothesis that this dissertation tries to prove. The evaluation conducted in this research is not supposed to evaluate the quality of the resulting sessions with regards to their underlying assumptions, even more so, given that the majority of the tested identification approaches uses unsupervised mechanics defining similarity in very different and subjective ways. Basing the similarity between interactions on user traffic alone makes it difficult to evaluate the degree of similarity and the impact on resulting sessions, since there are too many unknown factors[26]. In the main, therefore, this dissertation does not compare the inherent quality of the sessions, but rather the impact of the different approaches on certain measures and use cases.

Therefore, this dissertation does not use a gold-standard. Instead, the evaluation is conducted by comparatively analysing all resulting sessions and applying a representative sample of all relevant approaches to a use case. The first step is to compare all session approaches to a set of common measures deemed to be representative of user behaviour and system usage. The current research offers qualitative and quantitative evaluation in the form of analysis with focus on broad comparison and specific examples. The second step consists of using a smaller subset of sessions in a cohort of business use cases. The impact of the resulting session on a downstream application may be an indicator of the inherent quality of the session algorithm as well, but focus remains fixed on capturing the differences between them. The evaluation process in this research is partly inspired by the work of Jansen et al. [113], Buzikashvili and Jansen [33], He and Göker [95] as well as Zhao et al. [294]. All the steps that were taken are now explained in more detail.

4.5.2 Case 1: Comparative Analysis

The first step in evaluating the impact of session-modelling approaches is to thoroughly analyse the resulting dataset. At first, all session approaches are compared with their respective variants in the same method category. Every approach is descriptively analysed and every variant is looked at and compared with the others. Following this, a comparison is made between all the methods and their variants and between their different approaches and sets of measures. These measures and the order of analysis are now explained.

[26] For example, marketing campaigns may strongly influence users and lead to a different sequence of visited categories.

The measures are chosen in the first instance primarily with the intention of highlighting their different approaches to representing user behaviour and the specifics of the system. The measures are supposed to show the differences in the assumptions behind the different approaches and their outcome. Two general areas are analysed: system performance and user behaviour with an additional check for the underlying assumptions for every session approach.

System performance is comparably easily measured. Several different indicators are described that are typically used for evaluating the everyday performance of a system. This includes the following:

- Number of sessions (total, per user)
- Conversion rate and bounce rate
- Number of events per session (total, per user)
- Number of lead-ins (total, per user)

The number of sessions is a standard key performance indicator (KPI) in e-commerce[27], often used in combination with other dimensions as well. The measure shows how an information system performs in general and especially over time. On a per-user basis it can give a hint as to how attached users are. They may come back often, regularly or on a variety of topics. The number of sessions over a time period and in total are important elements by which to compare the different approaches. They are fundamental; easy to calculate and revealing, they may already provide insight into how the different approaches and their variants work.

Calculating sessions over time is important in the case of this dissertation. All sessions are attributed according to the timestamp of their first interaction. This may provide interesting insights into how the differences between various session approaches are caused by their underlying assumptions. While a 30-minute temporal inactivity session is somewhat limited in scope, a logical session with no temporal boundary at all may span a whole year. This may result in very different distributions of sessions over time depending on the respective approach. The hypothesis here is that all sessions with a somewhat limited scope are more uniformly distributed over the year while the rest may have a slight tendency to accumulate in the earlier parts of year. This is because while approaches with no time constraint can span multiple days or even months, they only get counted as one session from their start day.

This also influences the conversion rates. Conversion rates measure the performance of a system in terms of the fulfilment of the business goal or any other transaction. This dissertation focuses on lead-outs as the primary business goal, since all the other mentioned goals of the business are either underrepresented in the data or the data lacks quality. Therefore, the ratio between the number of sessions and the number of lead-outs is calculated by dividing the number of sessions by the total number of lead-outs in a

[27] `https://support.google.com/analytics/answer/1032796`, retrieved 28 November 2021.

session. In contrast to common other calculations, this research utilizes the total number of lead-outs instead of a binary encoding (transaction happened or not) because the total number is a more direct representation of the fulfilment of the business goal. This may lead to conversion rates over 100%.

The bounce rate, on the other hand, is more independent. Bounced sessions are sessions with only one interaction – this is a simplified way of seeing a situation where in this one interaction, the user did not visit to purchase but to fulfil a certain information need in one single interaction. Still, the bounce rate captures the number of sessions with only one interaction, in comparison to all identified sessions, to provide a somewhat qualitative perspective. Since the bounce rate is used to quantify the quality of a system's contents[28], it is considered an important measure. Identifying a high number of sessions with only one click will give a completely different impression of a system's performance than a low bounce rate.

The number of events per session is another indicator of a well-performing information system (at least for some types of information systems) and is directly connected to the bounce rate. For an online price-comparison website, the number of events does not necessarily need to be high to be indicative of a functioning business case. The number of events per session can be compared to other dimensions as well.

The number of lead-ins is another factor to consider. Technically, this number measures system performance as well as user behaviour: it represents the number of times a user enters the system from an external source. A lead-in is therefore a new entry to the information system, basically characterizing all interactions that do not have a **http_referer** starting with the idealo parent domain. This measure is representative of the system usage in combination with user behaviour. It leaves room for interpretation and different assumptions as to why a user may enter the system frequently: if the system does not provide easy or enjoyable navigation (depending on the number of interactions); because s/he enjoys using it (also depending on the number of interactions); or a user may enter the system frequently because the system's marketing is good, with comprehensive coverage on search engines, for example.

After all, user behaviour is more diffuse than measuring system performance. Still, similar measures can be applied. Strong focus is on a per-user basis, but the general direction of measures is very similar. The goal is to gain an overview of how a user behaves when interacting with the information system and, in the best case, to create an estimation of their satisfaction. Several measures belonging to different fields were collected:

- Sessions per user and interactions per session
- Time measures (time spent on the system, time between interactions)
- Number of distinct categories, products and queries
- Visited page sequences

[28] `https://support.google.com/analytics/answer/1009409`, retrieved 28 November 2021.

The number of sessions and interactions per session per user are standard measures. For mechanical sessions, the assumption is that there are more sessions per **user_id** compared to logical or combined approaches because the mechanical boundary is simply more restrictive compared to the other mechanics that have been tested. The differences here will give insight into the nature of the approach; they represent a snapshot estimation of user behaviour on a system.

For the time measures, several calculations were made. For one, the minimum and maximum timestamps were stored per **session_id**. With these, calculation of time per session can help estimate the difference between potentially longer lasting logical sessions and mechanical approaches. Also, the average inter-interaction time was calculated on a per-user basis, helping to gain a view over whether there is a difference between the approaches here. The hypothesis here is that mechanical sessions will have relatively short and consistent inter-interaction times due to their inherent nature; the maximum inter-interaction time is defined by their session-identification mechanism. For example, a 30-minute temporal inactivity session will have a maximum inter-interaction time of 30 minutes. For logical (and combined approaches as well), these times are assumed to be higher and more variable, at least for the interleaving comparison contexts. The consecutive approaches may have the shortest inter-interaction times, assuming that users will work on the same information need in short bursts before switching to other topics.

The number of categories, products and issued queries visited per session per user provide other strong indicators of user behaviour. A high ratio of root categories, categories, products or queries may reveal a lot about different topics per session, while a low ratio will likely mean a specific topic or focus during this session. The hypothesis is that mechanical sessions tend towards a higher number of different topics while logical and combined approaches tend to be slightly more focused. This may well prove to be wrong due to the different comparison contexts and methods that relate to the latter. The same is true for the number of interactions.

Another element to consider is path analysis. The visited pages are saved in a sequence structure in chronological order with an additional mark for lead-ins. By performing a path analysis, often-repeated patterns can be identified that may be descriptive for the respective session approaches or user cohorts. This helps in the understanding of user behaviour by showing the way a user approaches the information system as well as to better understand the way the session approach connects different interactions and interaction sequences. The hypothesis is that there are strong differences between mechanical and logical approaches in the overall sequences but not in the sub-patterns of said sequences. Considering this, and to estimate the underlying assumptions for every session-identification approach, additional measures were collected. An important element to consider when comparing the differences between mechanical, logical and combined approaches is the degree to which they have fulfilled their underlying assumption. For mechanical sessions, this means they should have a reduced quantity of potential topics. For logical and combined approaches, the same is true; here, interleaving behaviour or, rather, very short sessions ought to be identifiable. The following measures are calculated:

- Number of (distinct) **root_category_ids**
- Number of distinct potential topics
- Number of breaks for ac and ad comparison contexts

Calculating the number of visited **root_category_ids** is the simplest way to estimate the same information need. A low number is estimated for every type of session, either mechanical, logical or combined. Additionally, the *u2vcac* logical session approach is used as a baseline for estimating the number of potential topics[29] in all the other session approaches. The *u2vcac* approach was chosen as the baseline as there's no specific way to measure potential topics aside from the **root_category_ids** and it stands as the most natural logical approach. In this context, 'natural' means that there is no arbitrary boundary to determine similarity but for an empirically defined formula that at least somewhat orientates towards the $\cos\varphi$. Furthermore, it acknowledges interleaving behaviour because of the comparison context and looks at the complete session history. This helps in identifying the number of topics in consecutive session approaches like the mechanical session-identification approaches.

The other element to explore is the extent of interleaving behaviour. For logical sessions, interleaving behaviour is expected. To identify potential interleaving behaviour, the number of session breaks are counted for all session approaches with the potential for interleaving behaviour. In this context, the term 'session break' does not mean that the session ends but that another session or an interaction belonging to another session has interrupted the respective session. The actual measurement, therefore, relates to the number of gaps between the same **session_id**.

Overall, these 17 measures[30] will provide an insightful overview of how the 135 session-identification approaches impact system performance and interpretations of user behaviour. They will reveal how the different approaches produce different outcomes, eventually leading to varying interpretations of certain assumptions regarding the behaviour of these measures. The analysis below will compare all different approaches, mechanics, variants and comparison contexts with each other by looking at totals, averages and standard deviations.

4.5.3 Case 2: Using the Data for Implementation

The second step in evaluating the session-identification approaches is to apply them in a practical setting. Practical setting means that all session approaches are used in a case study that may be of relevance in an e-commerce information system. This is modelled

[29] The term 'potential topic' is used because there is no way to ensure that the *u2vcac* identifies consistent and related topics, therefore this is measure is just an assumption on topic engagement.

[30] Overall: sessions, conversion rate, bounce rate, most frequent sequences. Calculated per user, averaged by approach: sessions, number of interactions per session, conversion rate, bounce rate, lead-ins, root categories, categories, products, queries, topics, time in session, inter-interaction time, interaction days.

as a component-level evaluation: all model parameters and input data will be kept steady while the structure of the input data is the only component that changes. The structure is defined by the session approach; every different approach will have an impact on the sequence representation of the input data. The results of the different tasks will then be compared and placed in relation to the respective session approach of the input data.

Doing so will ensure a consistent comparison base between the different session approaches. All the model parameters are kept the same for every experiment. The only component that will change is the session approach being used, which will have an immediate impact on how the respective data that is fed to the model looks. For example, all events of a certain **user_id** are taken as input data. The session-identification approach structures this input data depending on the session definition and the result is fed back to the model.

While there are many different use cases for the application of sessionized data in a productive environment, this research focuses on three specific applications to show the differences between the session approaches:

1 checking the integrity of category embeddings

2 inter- and intra-session-based recommendations

3 clustering of users based on their session behaviour

The first task is a reiteration of the procedure already tested in Section 4.4.3.3 in preparation for the logical sessions. The goal is to compare category similarity based on category embeddings on the session sequences with the structure of the category tree and with each other. Doing so will give a view on whether users visit similar categories (according to the category tree relationship) in their sessions. Based on the assumption that users will visit similar categories in one session, this would validate the system's category tree. This is a typical business case because a validated category tree benefits the navigation as well as SEO.

Observing different category similarities per session approach would indicate behavioural and structural differences in the sessions. To do so, the identified session sequences are applied again in a category embedding task to generate category vectors. The resulting vectors are then used again to get similar categories for every **category_id**. The difference between the resulting similarities of the per-session approach are then evaluated in the context of the category tree and in comparison with the session approaches.

For the second task, the goal is to recommend the user the right item at any given point of a session. The algorithm is supposed to predict what type of interaction comes next in terms of **category_id**. Depending on what pages a user visited and what page may come next, s/he may be more or less likely to perform a lead-out or an order, directly fulfilling a business case. Knowledge about the next **category_id** is equal to knowing whether a user may work on the same information need in the next interaction or were they to be gently guided towards another topic depending on their session and interaction history.

The third task is another relevant business case. Clustering users is commonly performed to understand the different target groups visiting the system and how they may be targeted by marketing campaigns to fully cater to their behaviour. Often, this is supplemented by adding information about how much financial value the user contributes or is likely to contribute to the system (to predict said value in the future) or by adding category affinities. In this dissertation, the clustering focuses on the generated sessions: users are clustered by certain session-related features using an unsupervised algorithm.

Preparation: Sampling

To be able tackle the tasks at hand, the first job was to reduce the volume of data. This involved sampling a suitable data subset from the original dataset. Technically, sampling is not strictly necessary, but it is usually the case that a great quantity of data does not necessarily improve the results of the algorithm; in the case of machine learning algorithms, the data is considered good enough when the accuracy rate of the learning algorithm is no longer improved by adding more data [151]. Sampling is a trade-off between accuracy and the efficiency of the calculation: samples that are too small lead to incorrect results or interpretations, data samples that are too large will drastically increase the calculation time [151].

This dissertation considers data from over 78 million **user_ids** with over 1.2 billion interactions from a complete year (2018). Feeding the total amount of data into any learning algorithm will be likely to result in very long computation times without making much difference to the result of the algorithm. With this quantity of data, though, it can be assumed that a decently sized sample will deliver sufficient results and also greatly reduce the run time. Therefore, a generously sized sample with similar properties as the original dataset should be valid to see differences in the outcomes of the different session approaches.

In order to calculate the necessary size of the sample dataset, several elements are important. Contrary to survey data, where the population size may not be known or the number of given answers may be incomplete, the dataset used in this research has advantages. First, the population is absolute: the number of **user_ids** in the dataset represents the actual total of users. Second, the data sample is big enough to ensure a high goodness of fit. In the end, this allows for a comfortable sample size without having to worry about the data not being representative of the dataset while using random sampling.

When calculating the sample size, therefore, it is important to consider the overall population size, the expected margin of error and the confidence level [109]. The population size is the total number of **user_ids** in the dataset (the number of rows in the dataset is not important and of no concern here). The sample should contain all sessions of a chosen user; the sampling, therefore, is performed on the user population, detached from the actual data. The level of precision was set at 0.01%; this percentage indicates the margin of difference expected between the sample and the overall population [249]. Likewise, the level of confidence was set at 99%; this percentage shows how close the sample is to the characteristics of the population [249]. Both values determine how large the sample size needs to be in order to decrease potential deviations from the dataset.

Cochran's [53] formula – the standard formula to calculate representative sample sizes – was employed to calculate the required sample size. The following equation was used::

$$\frac{Z^2 * p(1-p)}{e^2}$$

Z represents the z-score, the statistical value corresponding to the level of confidence. p is the target proportion in the population – related to the feature a survey would normally research – in this case set to 0.5 to get the biggest sample size possible. e is the margin of error, set to 0.5%. With these values, a minimum sample size of 66,564 is required. The formula is not directly applied on the dataset but on the user population, resulting in a recommended sample size of 66,505 (after correcting the value using the finite population correction [109]).

The true random sampling was performed using the Presto implementation[31] of Bernoulli Sampling [81]. The probability of being included in the sample was equal across each element of the population (i.e. the **user_ids**) [81]. The statement takes both the dataset as input and a percentage to indicate the fraction that should be the outcome of the sampling. The resulting dataset consists of the **user_ids** that should be in the sample. As Israel [109] mentions, it is common practice to add 10% to the required sample size. Therefore, the fraction required for the Bernoulli sampling was higher, a choice that resulted in 391,257 distinct **user_ids** with a total of 6,275,248 interactions in the sampled dataset. This ensured representability.

True random sampling on the individual **user_ids** was deemed a valid strategy considering the goal of this case study. The sampling is supposed to pick a number of users randomly from the overall population, where every **user_ids** stands the same chance of being picked. In theory, one could optimize the sampling (i.e. using stratified sampling or more advanced techniques) to regard other important variables; for example, the number of interactions, the number of sessions or even content measures like the number of visited categories. Since their impact on the resulting sessions (per session approach), which is important part here, is not entirely clear and cannot easily be assumed, proper stratified sampling would be out of scope since it would involve many more steps. Additionally, this dissertation does not optimize for a specific variable, so a true random sample on the **user_ids** is the most reasonable solution, ensuring a proper distribution of all variables.

While the resulting sample consists of the same columns as the original dataset, it only takes a selection of a different session approaches into account[32]. In view of the fact that the lower inactivity thresholds appear to create too many small sessions, they were not evaluated any further. As a substitute for having smaller logical sessions, the combined approaches with a 30-minute timeout seemed to do a better job, having a comparable

[31] Compare `https://docs.aws.amazon.com/athena/latest/ug/select.html`, retrieved 28 November 2021.

[32] Many of them performed similarly, so a selection is reasonable.

number of sessions as the **ti5** sessions. As for the other timeouts, one variant per comparison context was chosen. The calculated cut-off comparison method was again chosen to ensure comparability with the purely logical approaches. In addition, examples of the geometric sessions and the lexical combined approaches were taken into account as well. Overall, the following variants were considered:

- Mechanical: ti30, ti180, tfd
- Logical: u2vccc, u2vccd, u2vcac, u2vcad, ladb1, lcdb1
- Combined: u2vcti30cc, u2vcti30cd, u2vcti1cc, u2vcti1cd, u2vcti1ac, u2vcti1ad, u2vcti14cc, u2vcti14cd, u2vcti14ac, u2vcti14ad, lti1cdb1, lti1adb1, lti180adb1
- Geometric: geomu24cc, geomu24cd, geomu24ac, geomu24ad

The diversity of these examples should be sufficient to reveal the difference between the methods and underlying mechanics. The different variants are taken with care. The mechanical variants are obvious: the **ti30** variant is the industry standard and serves as a baseline for these types of sessions while the **ti180** sessions are chosen as a different variant of timeouts. The **tfd** sessions, with all interactions of a user per day combined into one session, are the simplest of the variants and are intended to show whether the level of simplicity is good enough in these scenarios.

The reasoning for the logical approaches are similar. The **u2vc** approaches with the dynamically calculated cut-off were tested within all comparison contexts. The simple lexical sessions serve as a baseline. For the combined approaches, taking the mechanical timeout and topical comparison together, a variety of timeouts were tested within all comparison contexts. The lexical comparison method still serves as the baseline here. Finally, taking an example from the literature, the geometric approach was employed, once again within all comparison contexts.

All experiments were run locally on a system using Python 3.7.2 along with the Gensim[33], TensorFlow[34] and the scikit-learn[35] libraries. The system was run on a macOS Catalina (10.15.7) with 16 Gb of RAM and a Radeon Pro 555X 4-Gb graphics card. All experiments were run multiple times to ensure stability of the results.

Task 1: Category similarity via sequence embeddings

The intention of the first task is to show structural differences between the session approaches. Divergences across the category's similarities mean that the session approaches identify structurally different sessions, indicating different user behaviour and overall Divergences in the structure of these sessions. Here, the business case can be described as validating the category tree: the general assumption being that users will visit similar categories in one go. Note that this business case is based solely on the assumption that users deal with about one topic per session (as per the commonly held definition of mechanical

[33] `https://radimrehurek.com/gensim/`, retrieved 28 November 2021.

[34] `https://www.tensorflow.org/`, retrieved 28 November 2021.

[35] `https://scikit-learn.org/stable/`, retrieved 28 November 2021.

sessions). Similarity is first and foremost indicated by the same **root_category_id** – having the same root category is a strong argument for an apparently high similarity (although not necessarily true). Using the sessions, the category tree can therefore be evaluated using the resulting similarities. A high number of categories with the same root category in the top similarity categories would be an indicator of a well-functioning category tree.

Although this business case is somewhat abstract, in practice, system owners would be able to use the calculated category similarity to improve the internal structure of their system. In this dissertation, a theoretical methodology to start such a use case is given. The use case described is not so much a real test of which sessions are likely to be the most like the category tree, but more an observation of how the various approaches will result in different similarities. The basic premise for this use case is that the identified sessions deal with one topic; the use case is dependent on that assumption. This is not necessarily true in practice, but it is the assumption for all the approaches used here. Therefore, it is assumed that all the identified sessions deal with a limited number of categories. Different root categories (and different similarity scores) indicate a more diverse session structure and therefore either errors in the category tree or incorrectly identified sessions. For now, the former is assumed to observe the results in this use case. This use case does not attempt to replicate a practical application, rather, it describes the methodology to do so and observes the differences in the results while doing so.

Again, the Gensim word2vec was implemented to calculate the category similarities. The task follows the previously explained procedure, using the same steps for preprocessing the data. This included removing repeated consecutive **category_ids** as well as filtering sequences to include only those that had more than one interaction. The minimum sequence length is therefore two **category_ids**. All session sequences were then fed into the embedding algorithm to calculate category similarities for every one of the 2,300 **category_ids**.

Table 4.4 conveys the selected session approaches, the number of sequences and the total number of tokens (i.e. **category_ids** in interactions), plus the vocabulary size (i.e. all unique **category_ids**) to show how changes to the inputs of the algorithm alter the various session approaches. In the main, the cause of these differences can be attributed to the preprocessing (whereby sessions that tend to identify as shorter are more likely to have a lower number of sequences and tokens). The preprocessing also causes a difference in the vocabulary size, in that once the repeated consecutive categories have been removed only one **category_id** remains. This may lead to more single-event sequences for the logical approaches dealing with exactly one topic, ultimately, therefore, also potentially resulting in a smaller vocabulary as seen in the combined approaches with a 30-minute inactivity timeout.

To mitigate the smaller corpus size and to account for the smaller sequences, 15 training iterations were made per run with a smaller window size of five (in comparison to the complete user history of the baseline). Otherwise, the hyperparameter settings were identical to those used before. All runs were performed on the full sample data per session approach; splitting the data into training- and test data or cross-validating the results would not be reasonable in this scenario, considering that the evaluation focuses on the actual results and not a downstream application. Once the categories and their similar categories had been prepared, the **root_category_ids** were added and the analysis was performed.

Approach	Sequences	Tokens	Distinct categories
tfd	316,734	1,076,315	2,331
ti180	307,201	1,019,712	2,331
ti30	293,872	941,603	2,330
lcdb1	275,360	958,983	2,310
complete history	254,190	1,665,652	2,334
ladb1	253,292	1,189,393	2,322
lti180adb1	248,812	1,171,349	2,322
geomu24cd	240,470	768,569	2,324
geomu24ad	235,259	770,745	2,323
geomu24cc	230,472	783,240	2,325
geomu24ac	226,155	793,745	2,324
lti1adb1	225,168	759,979	2,311
lti1cdb1	223,410	713,872	2,308
u2vcad	203,114	691,801	2,308
u2vccd	185,642	555,248	2,300
u2vcac	180,188	726,461	2,311
u2vcti14ad	179,607	594,318	2,305
u2vcti14cd	177,111	529,982	2,299
u2vccc	175,538	570,242	2,300
u2vcti1ad	169,494	512,774	2,299
u2vcti14cc	167,742	543,720	2,300
u2vcti1cd	165,899	479,777	2,295
u2vcti14ac	163,764	618,459	2,309
u2vcti1ac	161,062	525,641	2,300
u2vcti1cc	159,142	490,092	2,296
u2vcti30cd	156,673	434,272	2,292
u2vcti30cc	152,203	442,238	2,293

Table 4.4: Sequence length and number of items for word2vec.

The evaluation is split into two parts and is based on analysis of the top 25 similar categories per each of the about 2,300 **category_ids**. In the first part of the evaluation, the number of categories with the same root category in the top 25 similar categories (per category) were counted as a measure of its closeness to the category tree. All measures were averaged per session approach. While the top 25 was chosen somewhat arbitrarily, it was deemed good enough as a comparison metric. A high number of root categories indicates a more diverse set of visited topics in the sessions per session approach, whereas a low number indicates topical proximity. Note that this is not necessarily true, but allows the differences to be observed.

The second part of the evaluation checked for structural differences in the resulting similar categories. A high number of differences indicated a high structural difference in the sessions. In this part of the evaluation, the calculated similarity scores were observed in the form of the distribution of the cosine similarity per top 25 similar categories and per all similar categories per **category_id**. Furthermore, as the calculated cosine similarity is dependent on the input data, it also hints at how the input data influences the outcome of the algorithm [40].

A step-by-step example can be described as follows. First, the embedding algorithm is trained on the preprocessed session sequences from all session approaches separately. Next, per session approach and per **category_id**, the top 25 similar categories are calculated along with their cosine similarity score as well as the associated **root_category_ids**. For example, for the the category *Smartphones* (mobile phones), the top 25 similar categories, their **root_category_id** and their cosine similarity are calculated. The resulting list of similar categories with respective root category and their distribution of cosine similarity is now analysed per session-identification variant. For the example *Smartphones* (mobile phones), this would result in 25 different lists of similar categories per session approach, which are now compared to each other based on the resulting similarity scores and their structural composition. The assumption is that different session approaches will lead to different distributions and different numbers of root categories in the top 25 depending on the approach family. This is then repeated for every other of the about 2,300 categories per session approach. The results of the analysis are aggregated per session approach and evaluated.

Task 2: Recommending what comes next

The second task amounts to a classic business case implementation: using machine learning to guide users to certain pages and recommend relevant products or categories to them. These recommendations are intended to get the system's users to fulfil business goals faster or more securely, for example, placing an order during a session or showing them a product that makes an order more likely considering their (current) interests. A similar scenario could be used to make a forecast model of how the system is going to perform in the future. Since recurrent neural networks are the state of the art in this type of task [102, 225, 250, 256], an already-tested variant from the literature was reproduced. The methodology decided upon for the current dissertation reproduces the approach taken by Ruocco et al. [225], which focused on improving session-based recommendations by including information about previous sessions of a user along with several other baselines.

Using recurrent neural networks (RNN), Ruocco et al. formulated their task as a recommendation problem: thus, for every step (i.e. interaction) in a session, they provided a list of recommended items. In treating this recommendation problem ultimately as a classification problem, they then output a score of zero for all unrelated items and a score of one for the target item (i.e. the item (**category_id** in this case study) with which the user will interact in the next step of the sequence). Furthermore, alongside their own approach, they tested a solely session-based RNN (based on [102]) and multiple non-sequential baselines, such as using most popular items, most recent items and a k-nearest Neighbors (kNN), per item, for recommendations. Both the details of implementation and original settings have been closely followed and reproduced in this dissertation[36]. The two RNNs as well as two of the baseline algorithms tested in the original article are also put to the test in this dissertation (most popular and knn).

Implementation of additional parameters – focused on the session length and session structure – to ensure the model worked efficiently, were considered and then adopted. As an example, one step in the preprocessing of these parameters removed all repeated consecutive interactions; e.g. multiple interactions by one user with a category will only show as one interaction in the input data. The purpose of this is to prevent the recommender from making recommendations for the same category while the user is actually interacting with the said category. This is reasonable considering the nature of the algorithm and the data at hand; otherwise, the most likely prediction or recommendation for the next step would probably always be an item repetition.

Another step is to cap the session length at a maximum of 20 steps. An input sequence will therefore contain a maximum of 20 interactions before the sequence is split into multiple sequences (to avoid losing any information in longer-running sessions) up

[36] Compare `https://github.com/olesls/master_thesis/`, retrieved 28 November 2021, for the original code.

to a maximum of 40 interactions. While this session cap potentially interferes with the logical sessions modelled in this dissertation (because they tend to be longer than the usual mechanical session), nevertheless, having carefully analysed the percentiles of session interactions per session, it was decided that a cap of 20 interactions was reasonable[37]. As a last step, only users with at least three sessions were taken into account for training and testing the models. This was a necessary step in order to have at least one additional training session for the inter-intra-session-based RNN. While the last step can be seen as a clear limitation when it is considered that the majority of users make only a limited number of interactions, it seemed reasonable within this use-case scenario.

The preprocessing decisions were left as they were designed in the original work by Ruocco et al. [225]. The other parameters were largely left as they were as well. For the embedding layer that is used in the original work instead of one-hot-encoding the item sequences, an embedding size of 100 was chosen for the dataset. The learning rate was kept at 0.001. The maximum number of recent session representations for the inter-intra-session RNN was set to 0.2. All models were run up to a maximum of 10 epochs after observing that the loss and evaluation score did not any longer improve dramatically.

It is positive that the algorithm may have reached a turning point to deliver better results, but since the sole purpose of this dissertation is to estimate the impact of the input data, additional tuning was not carried out. As proposed in the original work by Ruocco et al., the algorithm was trained on the training data and then evaluated on the test data: a split was placed between the sessions of a user instead of splitting the whole population. This means that all algorithms were trained on 80% of the first sessions of all users and evaluated on the last 20% of sessions per user.

Cross-validation would usually make sense in the current scenario to avoid overfitting and to be able to generalize the algorithm, making sure it produces stable and comparable results on new data, but this has been left aside in this dissertation for multiple reasons. For one, the original work does not use cross-validation, thus in order to reproduce it as closely as possible, the same procedure has been followed. Although this procedure (i.e. taking all sessions for training instead of the last one) has clear limitations[38], it was deemed good enough in the light of this dissertation – in this concrete use case, it is not important to have a high-quality algorithm. The goal of this case study is simply to prove

[37] The 90-point percentile for all session approaches was below 20 interactions, the highest being 16 for a logical approach. Considering that the first step was to remove repeated consecutive interactions (and thereby additionally shorten the sequences), the majority of sessions were included nonetheless.

[38] Other authors criticize this as well [156], but still reproduce it or adapt it only slightly. The problem with this task in the past is that these algorithms base their recommendations on a historic context; it may be that things become jumbled if sessions for training are split randomly over time because future sessions may involve interactions with new items, causing logical errors when predicting items in older sessions. Ludewig and Jannach [156] put forward the example of news article prediction, where recommending items in such a fast-changing environment might lead to inconsistencies. This is unlikely to be a big problem in the use case at hand, but is another argument for reproducing the original set-up.

that there are differences in the output caused by the nature of the input, therefore any effort to improve algorithm quality is not necessary, as long as stability is ensured.

The evaluation was done using recall@k and Mean Reciprocal Rank (MRR)@k using k values at 1, 5 and 20 to predict the next item in a sequence given all previous items in this sequence. With k the number of suggestions for the next step in this scenario, recall is the proportion of interactions per session step having the correct **category_id** among k in relation to all tested sessions at this step. The order of the ranking does not matter for the recall in this set-up. This, also referred to as Hit Rate (HR) in other works [156], was already used by Hidasi [102] and is reproduced by Ruocco et al. [225]. MRR is the average of all reciprocal ranks of the target item among the top k. Since the task is essentially evaluated as a prediction task, k@1 actually is just that: the MRR at position one equals a correct prediction of the next interaction. The latter represents a new addition by this dissertation: knowledge about the extent to which the algorithms are able to correctly predict the next **category_id** per session approach input is an interesting measure in terms of the nature of the different session approaches. The expectation is that a huge divergence will be seen here between mechanical and logical sessions.

In the current scenario, the evaluation strategy seems reasonable. The use case is an offline evaluation, without future access to further data to track the user's reaction, when given a set of recommendations. Given only the actual (historic) data at hand, it makes sense to assume that how the user reacts in the given sessions is the gold standard for the evaluation, especially since the algorithm's performance is being measured by estimating how well it is able to predict the unseen items in a session [156]. From this perspective, it is reasonable to measure recall/HR and MRR as proposed.

It is possible to quantify the impact on algorithm quality through a comparison of the evaluation measures of all algorithms among the different session-identification inputs. No matter how good or bad the algorithm performance, the dedicated effect of the input data becomes visible through comparison of the outcome measures.

Task 3: Clustering users based on session behaviour

The final task is again a typical business case. Commonly, users are attributed with certain additional properties (i.e. behavioural engagement with the system) to predict their financial value in the future, thereby enabling the system to attribute certain costs to certain actions. For example, if a certain number of typical contactable users generate a certain amount of money in a certain time period, an email to this user group could be estimated to cost a certain amount of money and generate a predicted amount of financial outcome. A common way to identify these user groups is through clustering the user population.

Since the aim of this dissertation is to show the differences between the various session-identification algorithms, the clustering is focused on session behaviour. The goal is to discover patterns in how users interact with the system as defined by the session approach. The clustering is intended to function as a way to explore the data from an algorithmic point of view, helping to identify patterns in the input data. If there are any noteworthy patterns as imposed by the selected input features, the cluster algorithm should find them;

otherwise, any respective data point should be not associated with a cluster. McInnes and Healy [172] describe this as the difference between clustering and partitioning: while clustering only finds naturally grouped subsets of the input data, partitioning associates every data point with a cluster, no matter what. In view of the fact that the current research has only sought to find out if there are clusters that would form naturally, data points with no likely association with a cluster should be considered as noise in the respective dataset.

In a set-up like this, without any labels or prior knowledge about potential clusters, the choice of the clustering algorithm is essential. By not setting up any assumptions on the data beforehand, a minimal hyperparameter selection is important; the algorithm should be robust towards noise and give information about data points that cannot be associated with any clusters [172]. For this reason, this research employs a clustering algorithm well-suited to exploratory data analysis: Hierarchical Density-Based Spatial Clustering of Applications with Noise (HDBSCAN)[39], which is based on the works of Campello et al. [34] and McInnes and Healy [172]. Their algorithm is an improved hierarchical version of the well-known Density-Based Spatial Clustering of Applications with Noise (DBSCAN) established by Ester et al.[74], allowing for clusters of differing densities calculated on varying epsilon values (the distance value used to determine clustered data points). HDBSCAN is ideal for the current task since, technically, it does not require any parameters beforehand, allows clusters of varying density and identifies noise in the data.

The input data for the clustering algorithm is defined by the session approach. Every feature is aggregated on a **user_id** level. This leads to every input dataset being the same size but with differing contents depending on how the session-identification approach structures the data. Considering that the selection of input features ultimately has a strong impact on the meaning of the identified clusters [99], the features were chosen rather strictly. As Hennig [99] states, the selection of features is directly dependent on the context and the clustering aims; therefore, a limited number of strictly session-related features were chosen to achieve the greatest impact on the features of the resulting clusters. As the number of sessions per user and the average number of interactions are the most descriptive measures when comparing the session algorithms, only these were taken into account. Looking at the broader picture, any average content measure is likely to dilute clusters when it is considered that the majority of users are assumed to behave rather similarly no matter the session context.

All features in the list were normalized using the MinMaxScaler implementation in scikit-learn [40] to ensure that they were the same scale. Since the effect of normalization seemed to have a great impact on the resulting clusters (as predicted [99]), the scaling was not changed for any session algorithm. Likewise, no principal component analysis was conducted, since this would again have an impact on the selected features, potentially leading to different clusters [99].

[39] `https://github.com/scikit-learn-contrib/hdbscan`, retrieved 28 November 2021.

[40] `https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html`, retrieved 28 November 2021.

Parameter-wise, HDBSCAN does not require many parameters to be set beforehand. One of the more important ones is the minimum cluster size, deciding on how many data samples should be included in order to form a cluster. Equally as meaningful is the minimum number of samples required in a neighbourhood for a datapoint to be considered the core point of a cluster. Both parameters have a great effect on each other and the resulting clusters. Since the default values[41] seemed to deliver very poor results in terms of identified clusters, both parameters were minimally tuned to better fit the dataset. The dataset contains 391,257 rows. The minimum cluster size, therefore, was somewhat arbitrarily set at 10,000. A cluster should only be formed if it contains 10,000 **user_ids**. Likewise, the minimum number of samples in a neighbourhood was set to 1,000 samples. Having the default values led to very high numbers of clusters with a lot of samples regarded as noise; setting the parameters to slightly higher values seemed to have a positive effect on some example runs in terms of resulting clusters and number of noisy samples. It is important to note though that changing these parameters also has a noticeable effect on the silhouette score of the different session approaches; apparently, changing them leads to different scores. This is already an indicator of different patterns in the data that would need treating differently to find proper user-behaviour clusters were clustering actually performed.

The evaluation of any resulting clusters identified by the algorithm is somewhat problematic. Again, no cross-validation or train- and test split is needed, since this use case does not optimize for algorithm performance. No quality indicator for training purposes exists at present. As Hennig [99] discusses, there are many caveats and a variety of factors to consider not only when choosing the clustering algorithm, but also when preparing the input data and measuring the outcome of the algorithm. This dissertation does not try to come up with the best possible way to evaluate a cluster analysis on **user_id** data. Rather, as stated, its aim is simply to show the differences in outcome of the various session approaches. This is an important consideration; for a cluster analysis, a great number of elements have an impact on the resulting clusters. From the choice of algorithm and respective parameters over the chosen preprocessing in terms of normalization and transformation up to the feature selection: every choice may result in different clusters. This also means that the evaluation measures are not independent from these choices: depending on the desired outcome, the evaluation measure may be less or more prone to bias influenced by previously made assumptions. Ultimately, the aim of this dissertation is not to validate the discovered clusters. Therefore, any measure taking cluster structure into account will do.

The results, therefore, are evaluated pragmatically and are not intended to determine cluster quality. As the most basic comparison measure, the silhouettes score[42] was calculated for all session approaches and their resulting clusters. The silhouette score is one way

[41] Refer to the API documentation for further information: `https://hdbscan.readthedocs.io/en/latest/api.html`, retrieved 28 November 2021.

[42] `https://scikit-learn.org/stable/modules/clustering.html#silhouette-coefficient`, retrieved 28 November 2021.

of measuring the average distance between a datapoint, all datapoints in the same associated cluster and all datapoints in the next nearest cluster. The results are in the range [-1, 1], indicating incorrect clustering or highly dense clusters. Technically, this indicates cluster quality, but this dissertation simply uses the measure as a way of showing different results for differing input data on a stable set of parameters. In addition, the number of resulting clusters is discussed as far as reasonable.

This concludes the chapter about the research design in this dissertation. First, multiple concepts and definitions were presented. This was followed by sections outlining the steps that were necessary to create a valid dataset, implement and develop multiple session-identification approaches and then how to evaluate these using analysis and machine learning tasks. Chapter 5 now evaluates all the approaches discussed in this chapter.

Chapter 5

Analysing Session Concepts

This chapter discusses and thoroughly analyses the outcome of the approaches explained in Chapter 4. The first section takes a closer look at the dataset in general and its inherent data model, performing an exploratory analysis. Afterwards, the different session-identification algorithms are analysed, visualized and compared. The last section discusses the results of this analysis.

5.1 Descriptive Analysis

This section takes a deeper dive into the dataset. With the knowledge reported in Chapter 4, the following analysis will look at the data model and explore what special properties could affect the session identification. An exploratory overview of the different aspects is given. The dataset contains interaction data for the complete year of 2018. One row corresponds to one interaction, with a total of 1,268,619,378 interactions from 78,361,923 **user_ids** on the German web platform idealo, but the app is excluded.

The first step to understand the data is to explore the basic statistics. To get an overview of the traffic distribution over time and especially per **user_id**, Figure 5.1 shows the distribution of interactions and **user_ids** over the complete year. The distribution of all interactions shows a noticeable seasonality towards the end of the year. This is predictable behaviour, because the e-commerce seasonality cycle usually begins in November for the start of Christmas shopping. November has the most interactions as Black Friday is a massive traffic driver, followed by December with another spike for Christmas shopping. Apart from these two months, the interactions are more or less equally distributed over the year. The distribution of distinct **user_ids** follows a similar trend. The similar pattern indicates a comparable behaviour among the **user_ids**: they come back regularly or there is a consistent share of new **user_ids** per day. In any case, there is no outlier here.

The traffic over the course of a day and over the course of a month is equally smoothly distributed. In Figure 5.2 and Figure 5.3, the hourly distribution of interactions per weekday and per day of month are shown. Both are calculated over the complete dataset, aggregating the interactions by hour and by weekday or day of month respectively. Figure

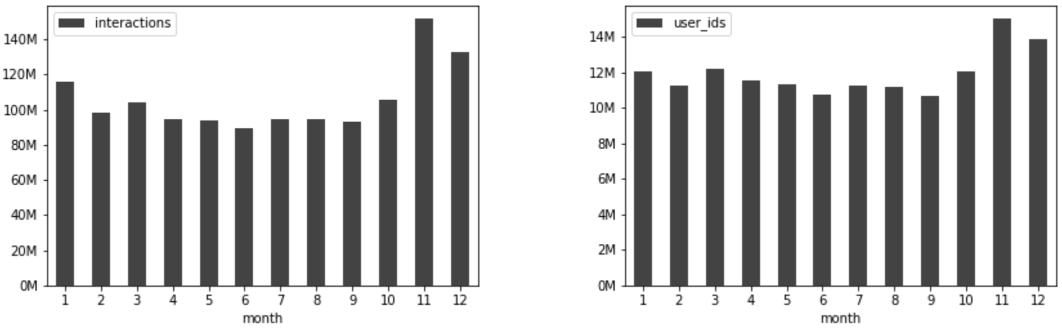


Figure 5.1: Distribution of interactions (left) and user_ids (right) over time.

5.2 illustrates the interactions in a heat map; the darker the shade, the more interactions were made.

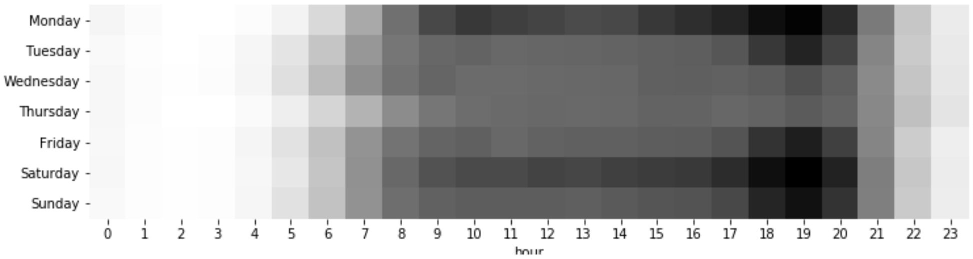


Figure 5.2: Heat map for weekdays and hours based on interactions.

The distribution per weekday has a slight tendency towards weekday evenings, normally in the later evening. This would make sense, as users are more likely to use the system in their leisure time, so that in general the majority of interactions happen after normal office hours or later in the evenings. Saturdays and Sundays tend to be the busiest days in terms of interactions as most users have more time to browse and shop than on other days. Mondays are strong as well – at first this seems counter-intuitive, but can be explained by specific shopping events such as Cyber Monday or Amazon's Prime Day. Another explanation for relatively busy morning hours especially on Monday could be free time on the way to work, although that should not only affect Mondays.

Over the course of a month, there seems to be a tendency towards the earlier days of the month. There is indeed an outlier here: the 23rd day of a month is heavily influenced by the Black Friday shopping event on the 23rd of November. Aside from that, the users seem to come more often in the first few days of the month which is most likely related to the receipt of their salary. There also seems to be a tendency to shop more often in the early evening hours and in the mornings, which confirms the impression from the distribution on the different weekdays. One assumption could be that these shopping sessions begin on the commute to work and on arriving at the office or begin on the commute home in the evenings after the working day.

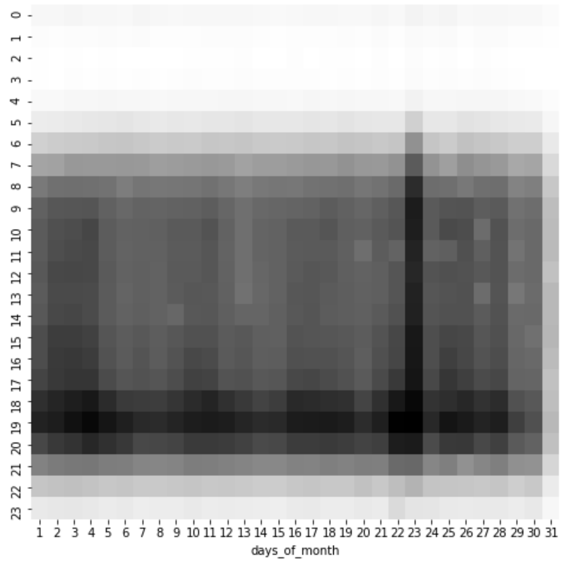


Figure 5.3: Heat map for months and hours based on interactions.

Following the distribution of interactions and **user_ids** over the year, it is assumed that the majority of the **user_ids** perform only a small number of interactions on the system. This is actually underlined by the data. The distribution of interactions per **user_id** is highly skewed, although already levelled out by removing the **user_ids** with only one interaction. The majority of users do indeed have a low number of overall interactions over the year. Looking at the overall number of interactions per **user_ids** as a dimension, it has a cardinality of over 5,000. When counting the number of distinct **user_ids** per number of interactions, there is clear picture: 90.76% of all users perform less than or equal to 30 interactions per year. The remainder of the 5,000 interactions accounts for roughly 10% of the total number of **user_ids**, the highest of those having over 20,000 interactions over the course of the year. These can be considered as artefacts despite that they do not seem to be traditional bots or crawlers, as their behaviour is not really consistent (i.e. no programmatic access, no consistent inter-activity time).

In Figure 5.4, a histogram with the distribution of the number of distinct **user_ids** per the overall **number_of_interactions** (per user) is shown in buckets of 10 up to a maximum of 100 interactions overall. All interactions above a total of 100 interactions are grouped into one bucket. As can be seen, the distribution here is fairly skewed as well. The majority of users performed less than 10 interactions with the information system, with a smaller subset making between 10 and 30 interactions. In terms of user behaviour, this could indicate that most **user_ids** use the website either in one comparably longer visit or in several visits with a small number of interactions respectively.

In Figure 5.5, the distribution of the first 30 interactions is shown. Here the curve looks a lot more level but is still skewed towards the lower number of interactions; the highest

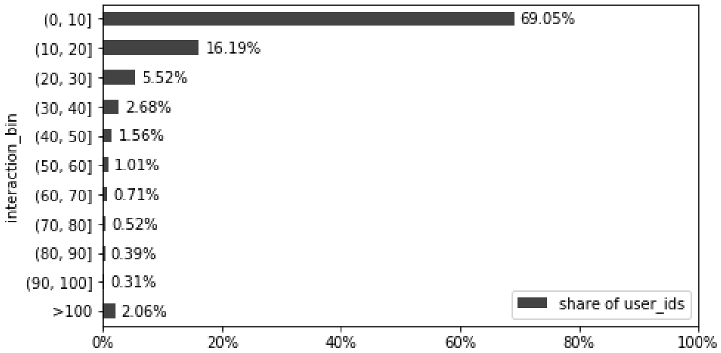


Figure 5.4: Details of interaction buckets: 10–100 and over 100 interactions.

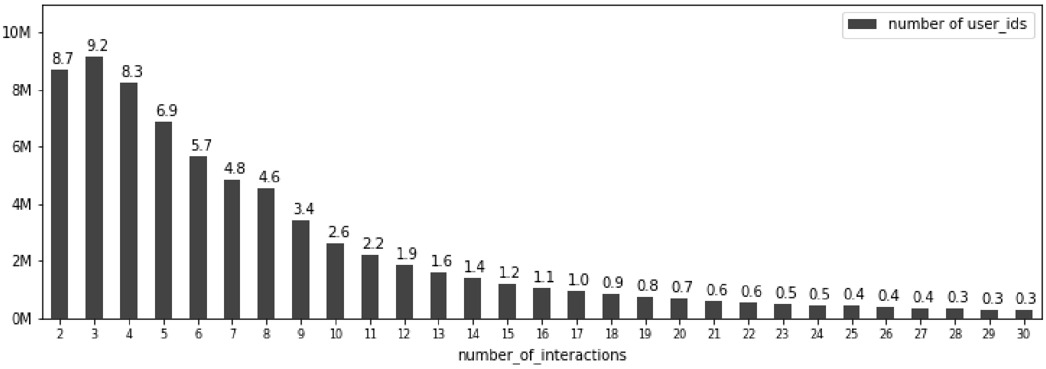


Figure 5.5: Details of overall number of interactions: 2–30.

number of **user_ids** only makes two or three interactions with the system over the course of the year. The impact of different session modelling approaches on this broad scope of users will be interesting. It can be assumed that the system-performance related measures in particular will be greatly impacted by the different approaches: with the majority of the user base making up only a small number of interactions overall, it may be crucial to the system to see how many sessions they end up with.

On the other hand, the actual big bulk of interactions are not performed by the **user_ids** with less than or equal to 10 interactions overall. The data shows that 90% of **user_ids** with less than or equal to 30 interactions only account for roughly 45% of all interactions; 50% of interactions are made by **user_ids** with up to or equal to 1,000 interactions; and 5% are made by **user_ids** with more than 1,000 interactions. This is an interesting finding. While most **user_ids** presumably only visits once or twice, the most interactions are made up from **user_ids** that are likely to come back more regularly (or make more interactions in a single long visit).

Therefore, the most interesting cohort may be the **user_ids** in-between these buckets: with more than 30 interactions and up to 1,000 interactions, this cohort provides a decent chunk of interactions with the potential of multiple different information needs and a more regular engagement with the system. In Figure 5.6, the difference between the buckets is

	Total		≤ 10		>10,≤ 30		>30,≤ 100		>100,≤ 500		>500	
share of **user_ids**	100%		69.05%		21.7%		7.18%		1.9%		0.18%	
share of interactions	100%		21.58%		22.91%		22.7%		21.95%		10.8%	
share of lead-ins	39.8%		51.7%		39.95%		38.52%		35.58%		26.9%	
	AM	**SD**	**AM**	**SD**	**AM**	**SD**	**AM**	**SD**	**AM**	**SD**	**AM**	**SD**
interactions	16	64	5	2	17	5	51	18	189	90	982	908
lead-ins	6.44	18.4	2.62	1.5	6.82	4.17	19.74	11.89	67.23	43.1	264.1	209.6
interaction days	3.41	7.27	1.64	0.97	3.6	2.72	10.2	6.7	31.4	18.8	97.96	50.6
visits	6.45	18.85	2.62	1.5	6.8	4.09	19.6	11.4	66.9	41.5	277.3	219.7
root categories	2.13	1.38	1.67	0.76	2.55	1.25	4.1	2.1	6.2	2.1	8.8	2.1
categories	3.64	5.57	2.12	1.1	4.1	2.46	9.6	5.5	24.8	13.9	68.7	40.8
queries	1.95	10	0.84	1.1	1.79	1	5.35	4.83	20.55	18.24	114.4	182.9
products	4.03	10.82	1.65	1.33	4.68	2.82	12.4	6.77	38.22	21.87	149.45	136.25
days between interactions	4.9	12.3	5.5	14.3	4	5.8	2.9	2.6	1.3	0.85	0.4	0.18
minutes between interactions	205	250	196	271	215	208	257	163.5	240.1	114.8	163	72.5

Table 5.1: Summary statistics for user cohorts (based on overall number of interactions). The first three measures are calculated as a share of all interactions. The rest is an average of the distinct count of the respective measure per user_id. Abbreviations: AM = Arithmetic Mean, SD = Standard Deviation.

shown again in terms of performed interactions and the number of **user_ids**. The number of interactions per cohort is calculated as the ratio to the total of interactions in the dataset. The same is done for the **user_ids**. The actual bucket boundaries are based on the number of interactions and how the respective buckets contribute to the overall number of events in the dataset.

The buckets have a very different share of **user_ids**, but the interactions made by these users are comparable overall. All the buckets except for the last one perform slightly more than 20% of all interactions, but the number of users is quite different. This is an indicator of variations in user behaviour. Having less **user_ids** in the bucket but more or a comparable number of interactions clearly indicates that with a higher number of total interactions, these **user_ids** are likely to come back more often or spend more time on the system; in any case, the **user_ids** in the higher interaction buckets make more interactions compared to those in the lower buckets.

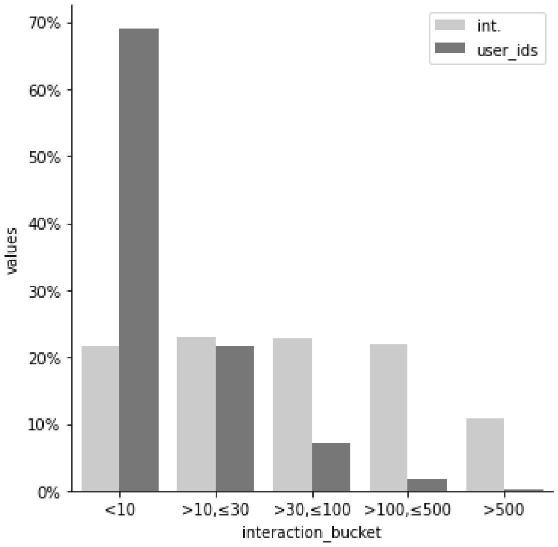


Figure 5.6: Difference in share of interactions and user_ids per interaction bucket.

Summary statistics for the buckets are depicted in Table 5.1. This table shows multiple statistics on an averaged per **user_id** basis, partitioned into several interaction buckets. The buckets are chosen according to the previously displayed cohorts: less or equal to 10 interactions; between 10 and less or equal to 30 interactions; between 30 and less or equal to 100 interactions; between 100 and less or equal to 500 interactions; and above 500 interactions. A total across all **user_ids** is also shown for comparability.

The first three rows display the ratio of measurement per interaction bucket. They show the number of (unique) **user_ids** per bucket per all (unique) **user_ids** in the dataset, the number of interactions per bucket in relation to all interactions in the dataset and the share of lead-ins per interaction bucket compared to all interactions in that interaction bucket. That means, as an example, that 39.8% of all 1,268,619,378 interactions are a lead-in, as shown in the column for the totals. The following rows contain different calculated measures on a per **user_id** basis and are averaged per interaction bucket. All these measures count the distinct value per **user_id** except for the last two rows which simply average the inter-interaction time per **user_id**.

At its most general level, the table provides information about the number of interactions as well as the arithmetic mean of interaction days and visits. The interaction days give an indication of the number of days a user enters the system. This enables an approximate estimation of how often or regularly the user visits the system. The number of visits looks at a similar theme but now at a more granular and independent level, as multiple visits may occur on the same day. A key to understanding the browsing behaviour that is connected to the relevant cohort most likely lies in the number of lead-ins. This measure allows an interpretation of how often the user returns to the site and how many interactions are performed during a single visit. The number of lead-ins is displayed per number of interaction cohort. The table also gives information about a lead-in per **user_id** ratio, showing how many of those may differ between the different users of a bucket.

The time between interactions is displayed as well, once in an abstracted way as the number of days between interactions and once in a cleaned abstraction in minutes. The time in days simply averages all the time spans between events, no matter how large they are. The time in minutes is cleaned: here, all time spans longer than 1,440 minutes are fixed to the same value. This has been done to give a more realistic although simplified picture of how the inter-interaction times look over the course of a day. Again, the inter-interaction times show the difference in system usage between the users of the different interaction buckets. This gives an even more granular take on how often and how regularly users interact with the system. The different times are also an explicit indicator of the potential for the later analysed mechanical session approaches: the longer the inter-interaction time, the more mechanical sessions depending on the chosen threshold. The interaction times can be viewed as closely connected to users' engagement with the system as well as user behaviour.

The other measures relate to the actual content of the pages visited and give an approximation of how users interact with the system overall: root categories, categories, queries and products. All of these topics relate in one way or another to the variability of topics a **user_id** engages with on the system. A high amount indicates a variety of topics while a small amount may hint at low engagement with the system, i.e. a low number of overall interactions.

Overall, the numbers are intuitively understandable and do seem to follow the hypotheses previously discussed in relation to the buckets. First of all, it is important to note the distribution again: The number of interactions is evenly distributed between the different buckets below a total of 500 interactions. The interesting part is how these interactions are structured. Considering that the bucket with less than or equal to 10 interactions is made up of the majority of **user_ids**, the hypothesis is that the lower the number of overall interactions, the higher the number of lead-ins are as well.

This at least in part is true. For the first bucket with less or equal to 10 interactions, the number of lead-ins is very high: 50% of all the interactions within this bucket are an entry to the system. Basically, this can be interpreted as follows: the majority of users are likely to bounce; they enter the site via a search engine or advertising, click maybe one more page and then leave until the next entry. Comparatively, for **user_ids** with only two interactions, the share of lead-ins is even higher at roughly 84%. This eases down a bit with the three buckets between 10 and 500 interactions. Here, a consistent ratio of roughly 35% to 40% of lead-ins seems to be the case – slightly lower, but consistently so, at a comparable value. This is an indicator that the behaviour stays at least somewhat similar in these buckets. The last bucket has a lower share, which makes sense considering that they have over 500 interactions over the course of a whole year. The total number of lead-ins calculated across all **user_ids** regardless of the number of interactions is similar to the middle buckets with 39.8%. This is another indicator that a share of around 35–40% lead-ins might be the standard behaviour.

This can also be understood by looking at the calculated averages per **user_ids**. The number of interactions can be seen as related to the number of lead-ins. For the lowest bucket, there is an arithmetic mean of five interactions with a standard deviation of two and a ratio of lead-ins of 2.62 with a standard deviation of 1.5. This underlines the previous point: the majority of these users enter the system on one or more occasion and performs a very small number of interactions. Looking at the same numbers for the higher buckets, a similar connection can be observed. The average numbers are higher and the relation between average interactions and lead-ins is slightly higher. The ratio between the buckets matches that of the overall ratio of lead-ins.

The interaction days and visits follow a similar trend underlining the general assumption, although with a noticeable difference between the buckets. The bucket with less than or equal to 10 overall interactions has a surprisingly low number of interaction days and visits. On average, users have 1.64 interaction days, meaning that a decent number of these users visits the system only on one or two days. The visits are equal to the lead-ins which makes sense considering that they measure a similar thing. The next bucket has slightly higher numbers: with an average of 3.6 interaction days, they come to the system on more days but apparently with similarly structured sequences. The trend continues: the higher buckets have more interaction days, meaning that users return more often. Apparently, the behaviour of users remains similar, otherwise the ratio between the days and the number of interactions would change more drastically. This is the same situation for the lead-ins.

This does not really change for the content measures. The number of root categories and categories is a direct indicator of user's engagement (or not) with the system. The correlation between the number of interactions and the distinct number of visited root categories or product categories is obvious. The more interactions a user has made, the more likely it is that s/he looks at a more diverse range of categories. This is also connected to the range or number of products, although the increase seems to be slightly steeper. The number of distinct queries follows the same trend.

Roughly 55% of interactions contain a **product_id**, no matter the number of total interactions. A difference can be seen for the number of distinct **product_ids** though; it seems to be the case that the more interactions a user has, the more often s/he visits the same product again and again. This seems reasonable, as returning users might use the system for watching a product or working continuously on the same information need, while users with a small number of interactions, who use the system only sparsely, will check for a product only once. The ratio of interactions with distinct products per overall interaction ranges from 32% for the first bucket to 15% for the last bucket.

Another take on these ratios comes in the form of actual content. Regarding the previously made assumptions towards logical concepts, it is important to look at the distribution of user behaviour in terms of visited pages, query topics, categories and even products. The actual content of the visited pages is important, especially when put into context with the buckets for the number of overall interactions of a user. If the majority of users with a low number of interactions are only looking for a small number of categories, logical concepts will not be effective. The distribution of traffic on the **page_templates** was already shown in total in Figure 4.3, but is now looked at again for the different interaction buckets. The distribution between the interaction buckets stays relatively uniform. There is no significant shift between the **page_templates**, which is an indicator that user behaviour remains similar no matter how engaged the user is.

The most interesting observation here is that the lower interaction buckets seem to visit the homepage (*MainProductCategory*) less often compared to the more engaged users. Users with more interactions also seem to visit the product page (*OffersOfProduct*) less often compared to the other buckets. The first observation makes sense: coming from a search engine to look for a specific product will most likely not result in an interaction on the homepage. In contrast, heavy users may visit the system directly, coming first to the homepage to start their interaction se-

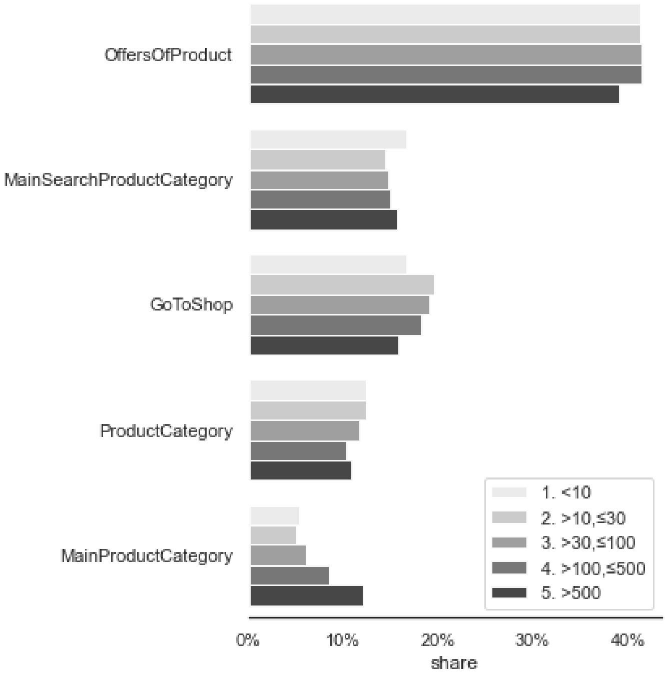


Figure 5.7: Details of interactions per page_template per interaction bucket.

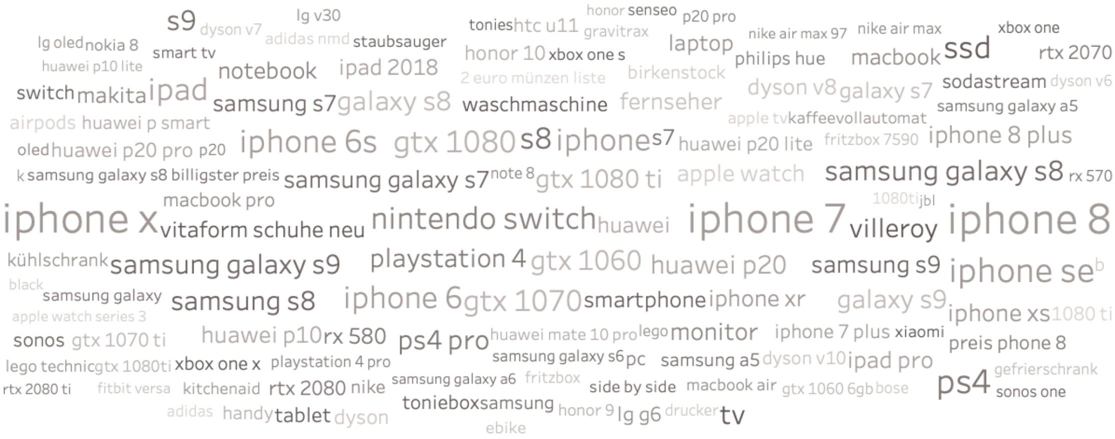


Figure 5.8: Word cloud including the top 150 most searched queries.

quence from there. The second observation is also reasonable; the more interactions a user makes, the more diverse the set of pages s/he visits will be.

Looking at the queries, categories or products with the most interactions by **user_ids**, it is possible to estimate the main topics users come with to work on the system. This is also important to get a feeling for the potential of topical connections between these identifiers. The majority of interactions happen in the root category *Elektroartikel* (electronics) (with close to 40% overall). Another 18% of overall interactions happens in *Haus Garten* (home garden). The remaining interactions are distributed more or less evenly among the others, with a small number of artefacts with a very low number of interactions. On a per-bucket level, there seems no be no real difference between the buckets.

The distribution of products and product categories follows Zipf's law with a more or less smooth curve. The most visited **category_ids** are *Smartphones* and *Fernseher* (TVs), followed by a long tail of all other categories. The distribution is similar for all interaction buckets. Except for the behaviour of some artefacts in the lower-interaction buckets, there is little difference: these have a tendency towards **category_id** *100* (indicating the search result page) and usually relating to one of the electronic categories. This makes sense as entry is usually via a search engine where these categories are promoted the most.

The focus on *Smartphones* is also visible in the queries, visualized in the form of a word cloud in Figure 5.8 for a snapshot view of the top 150 queries. Most of these belong to electronic products with different versions of the *iPhone* taking the lead. Other electronic products like graphics cards (GTX 1080) or video consoles (PlayStation 4) are also among the top queries. This is similar to the visited **product_ids** as well. The distribution follows Zipf's law. Again, there is no noticeable difference between the interaction buckets.

Returning to Table 5.1 discussion now turns to the measures relating to the inter-interaction time. Here, an interesting observation can be made. For the first measure (average number of days between interactions), the calculated values are intuitively understandable: the more interactions a **user_id** has over the course of one year, the shorter the time will be between these interactions in terms of days. A **user_id** with more than 100 interactions is likely to return to the system regularly while **user_ids** with less interactions may leave a longer period of time between individual visits. This indicator could point to another assumption: that users in the higher-interaction buckets return regularly whereas users with less interactions may leave a varying number of days in-between their interaction sequences. This is underlined even more by the comparably low-standard deviations in the higher buckets.

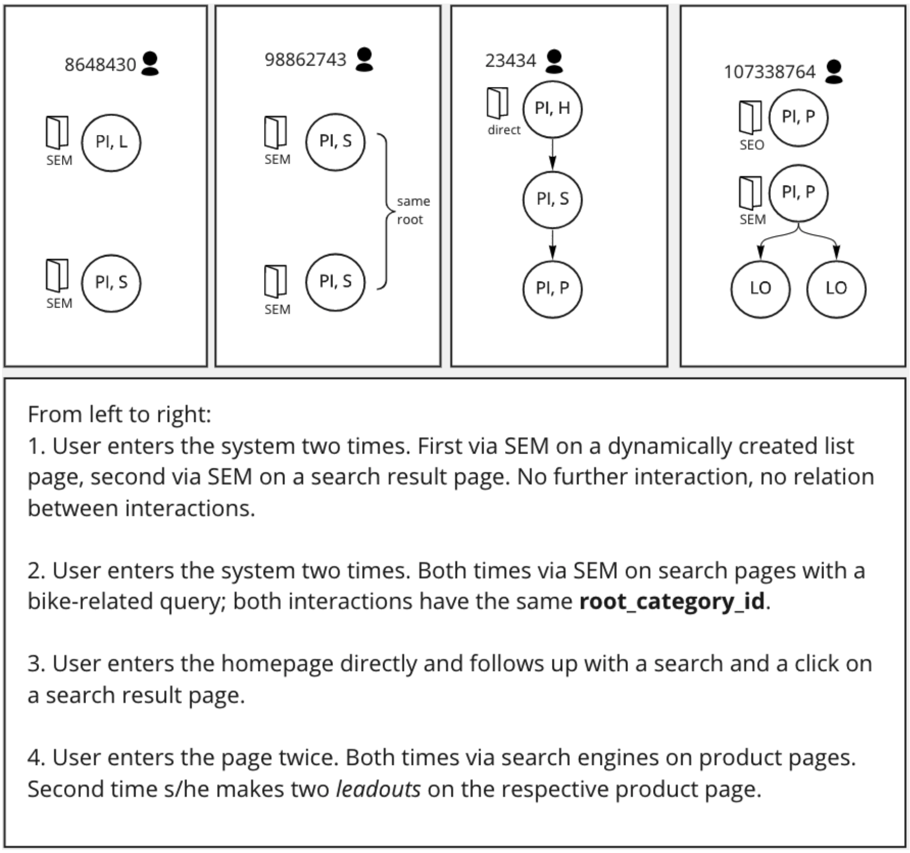
The same insight is suggested by the second measure in the form of the cleaned time between interactions in minutes. There are no huge differences between the interaction buckets. It can be assumed that the **user_ids** may spend a similar amount of time on the system with a tendency towards a normal distribution among the buckets: comparably, as the first cohort may bounce more often it may have a slightly lower inter-interaction time, whereas the last bucket knows the system very well and, therefore, also spends less time between interactions. The in-between cohorts have slightly higher time spans, but not significantly so on average. Here, the standard deviation is also not extraordinary.

This summarizes the descriptive overview of the dataset. There was no clear difference between the interaction buckets from various perspectives. Mostly, the users seem to perform comparable behaviour that only increases in frequency and partly in substance, especially with the more engaged users. Overall, there are some assumptions that can be inferred from this descriptive analysis:

- the frequency of interactions is higher the more engaged a user is, but the behaviour patterns are not that different (excluding queries)
- the number of visited categories is a good indicator of trust in the system, as the numbers increase the more users interact with it

Now, some examples will be presented and explained. These **user_ids** are chosen from the cohort because they performed an interesting series of interactions that may be representative of system usage. Here, the goal is to get a feel for typical user behaviour, detect differences or peculiarities in system usage and gain an understanding of how different shopping journeys may look. The figures use simplified abbreviations to show what the users are doing. The door symbol indicates a lead-in: the user enters the system. The circles indicate interactions. Arrows between the circles indicate a **url/http_referer** connection. Identical colours indicate the same **category_id**. The abbreviations in the circles describe the type of interaction and the page the user is on the time: PI equals *pageimpression*, LO equals *leadout*. The letters after the comma indicate the page type: L is the list page, S is the search result page (after a search), H is the homepage, P is the product page, PC is the category page and C is the cluster page.

The first batch represents a sample from the bucket with less or equal to 10 overall interactions with the system. In Figure 5.9, multiple **user_ids** with a very limited number of overall interactions are shown. As can be seen here, these users tend to come to the system via search engines and often stay for only one interaction. As the the statistics in Table 5.1 show, this behaviour is representative of a low number of overall interactions. Interestingly enough, these users seem to often visit the same category area. This contrasts



From left to right:

1. User enters the system two times. First via SEM on a dynamically created list page, second via SEM on a search result page. No further interaction, no relation between interactions.

2. User enters the system two times. Both times via SEM on search pages with a bike-related query; both interactions have the same **root_category_id**.

3. User enters the homepage directly and follows up with a search and a click on a search result page.

4. User enters the page twice. Both times via search engines on product pages. Second time s/he makes two *leadouts* on the respective product page.

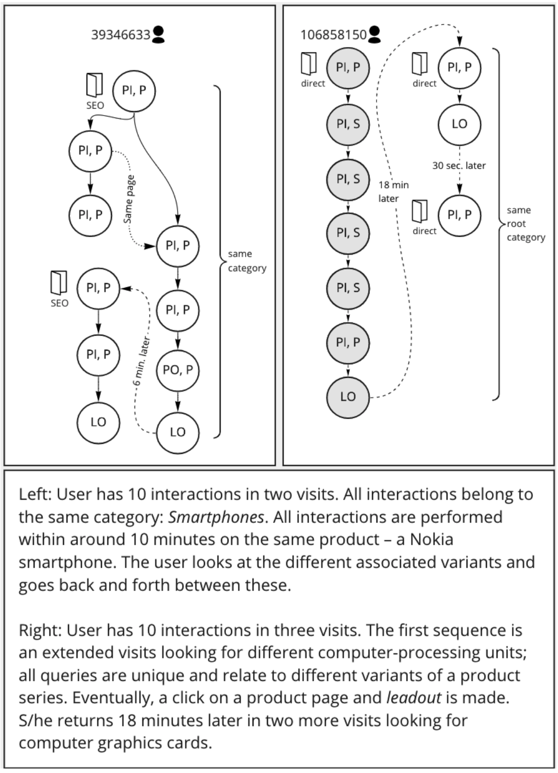
Figure 5.9: Clickstreams for four user_ids. Abbreviations: PI = Pageimpression; LO = Leadout; L = List page; S = Search result page; H = Homepage; P = Product page. PC = Product category page; C = Cluster page. The door symbolises a lead-in with the associated source. SEO = search engine optimization; SEM = search engine marketing.

with the two other examples, with a slightly higher numbers of interactions, pictured in Figure 5.10.

In these two examples users appear to interact quite differently with the system. They make longer sequences of related interactions in the same category range, but there are also the same short sequences as seen in the lower interaction bucket. This kind of mixed behaviour is a recurring observation. Figure 5.11 offers an example of interesting search behaviour where the user seems to heavily use the search, reload the page, and then return to the search result page to look for more and different results.

Another, similar example is shown in Figure 5.12. Here the user also performs a mixture of different sequence types, but also seems to be more engaged with the system, using different features like sorting or paginating through search result lists. This is an indicator for the higher interaction buckets.

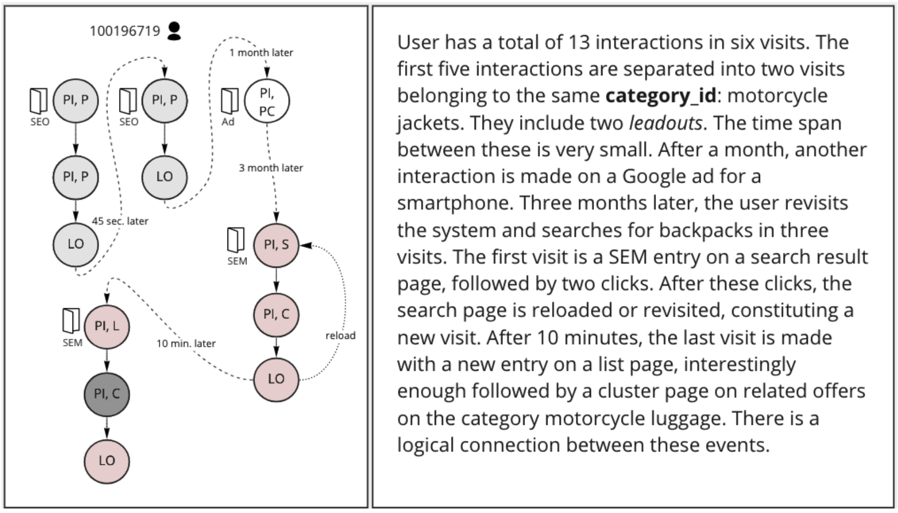
Displaying examples with even more interactions is cumbersome, but suffice to say that the behaviour in general is similar. As was explained in relation to Table 5.1, the more interactions a user performs, the longer the interaction sequences tend to be. More often than not, the tendency is for users to go back-and-forth to different product pages in a product category, and this is a behaviour frequently accompanied by searches or interactions on the system like sorting or paginating through result pages. Comparably,



Left: User has 10 interactions in two visits. All interactions belong to the same category: *Smartphones*. All interactions are performed within around 10 minutes on the same product – a Nokia smartphone. The user looks at the different associated variants and goes back and forth between these.

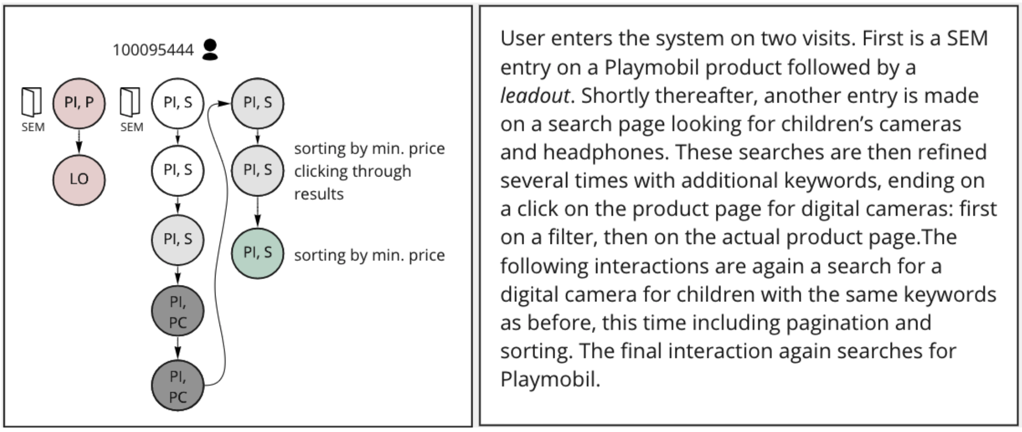
Right: User has 10 interactions in three visits. The first sequence is an extended visits looking for different computer-processing units; all queries are unique and relate to different variants of a product series. Eventually, a click on a product page and *leadout* is made. S/he returns 18 minutes later in two more visits looking for computer graphics cards.

Figure 5.10: Clickstreams for two user_ids. Abbreviations: PI = Pageimpression; LO = Leadout; L = List page; S = Search result page; H = Homepage; P = Product page. PC = Product category page; C = Cluster page. The door symbolizes a lead-in with the associated source. SEO = search engine optimization; SEM = search engine marketing.



User has a total of 13 interactions in six visits. The first five interactions are separated into two visits belonging to the same **category_id**: motorcycle jackets. They include two *leadouts*. The time span between these is very small. After a month, another interaction is made on a Google ad for a smartphone. Three months later, the user revisits the system and searches for backpacks in three visits. The first visit is a SEM entry on a search result page, followed by two clicks. After these clicks, the search page is reloaded or revisited, constituting a new visit. After 10 minutes, the last visit is made with a new entry on a list page, interestingly enough followed by a cluster page on related offers on the category motorcycle luggage. There is a logical connection between these events.

Figure 5.11: Clickstream for user_id 100196719. Abbreviations: PI = Pageimpression; LO = Leadout; L = List page; S = Search result page; H = Homepage; P = Product page. PC = Product category page; C = Cluster page. The door symbolizes a lead-in with the associated source. SEO = search engine optimization; SEM = search engine marketing; Ad: Advertisement.

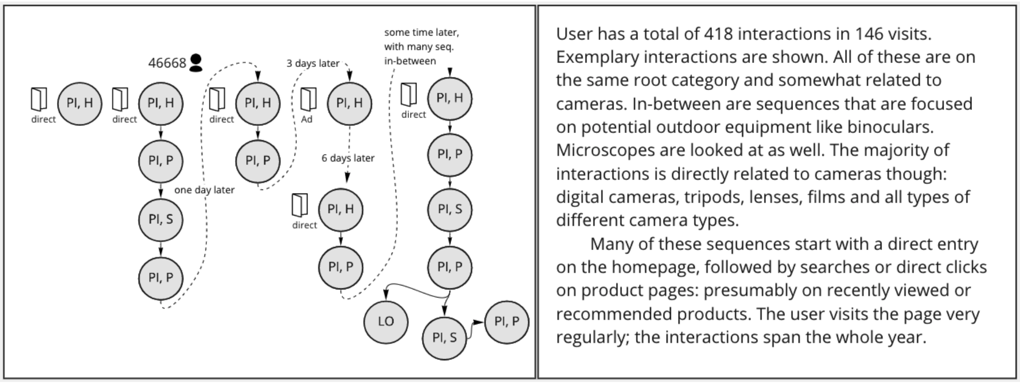


User enters the system on two visits. First is a SEM entry on a Playmobil product followed by a *leadout*. Shortly thereafter, another entry is made on a search page looking for children's cameras and headphones. These searches are then refined several times with additional keywords, ending on a click on the product page for digital cameras: first on a filter, then on the actual product page.The following interactions are again a search for a digital camera for children with the same keywords as before, this time including pagination and sorting. The final interaction again searches for Playmobil.

Figure 5.12: Clickstream for user_id 100095444. Abbreviations: PI = Pageimpression; LO = Leadout; L = List page; S = Search result page; H = Homepage; P = Product page. PC = Product category page; C = Cluster page. The door symbolizes a lead-in with the associated source. SEO = search engine optimization; SEM = search engine marketing.

the number of lead-ins remains the same, but the interactions following an entry are higher. The interaction sequences that are performed in one go are comparable to the **user_id** illustrated in Figure 5.12: several pages with several actions per lead-in.

A typical example of this is seen in **user_id** *46668*. With a total of 418 interactions, s/he belongs to the third interaction bucket and behaves accordingly. Around 34% (144) of all interactions are considered a lead-in. Looking at the paths the user performs on the system, multiple types of behaviours can be observed. On the left-hand side, the typical behaviour of a low-interaction user can be seen: several lead-ins with only a very few interactions (*pageimpression*, *query*, *leadout*) before either a long or a very short waiting time until the next lead-in. The other type of behaviour again resembles the paths seen in Figure 5.12: longer sequences with potential reformulations of queries or clicks that go back and forth between similar products or categories.



User has a total of 418 interactions in 146 visits. Exemplary interactions are shown. All of these are on the same root category and somewhat related to cameras. In-between are sequences that are focused on potential outdoor equipment like binoculars. Microscopes are looked at as well. The majority of interactions is directly related to cameras though: digital cameras, tripods, lenses, films and all types of different camera types.

Many of these sequences start with a direct entry on the homepage, followed by searches or direct clicks on product pages: presumably on recently viewed or recommended products. The user visits the page very regularly; the interactions span the whole year.

Figure 5.13: Sampled clickstream for user_id 46668. Abbreviations: PI = Pageimpression; LO = Leadout; L = List page; S = Search result page; H = Homepage; P = Product page. PC = Product category page; C = Cluster page. The door symbolizes a lead-in with the associated source. SEO = search engine optimization; SEM = search engine marketing; Ad: Advertisement.

Figure 5.13 displays a sample portion of the typical behaviour of **user_id** *46668*. Various lead-ins, often with only one click, are mixed with short and longer sequences of multiple different products and page types. User *46668* seems to be one of the heavier users, regularly coming back to work on the same or different information needs.

The interactions all belong to the same **root_category_id**. The **category_ids** are different between the sequences, but are all somehow related: the majority deal with *Photography* (which is also the root category), landing on different types of *video cameras*. Overall, the behaviour is consistent and intuitively understandable. The most interesting thing is the mix of behaviours, which is a good indicator that the behaviour between **user_ids** with a low and a high number of total interactions does not change very much, or at least not completely.

An estimation of typical user behaviour can be drawn from these examples in the context of the previously explained statistics. It seems that a common theme for all interaction buckets (and therefore all users) is working on a specific topic in the form of short sequences. These short sequences are best described as interaction bursts; users will come to the system for a limited number of interactions on the same topic before leaving again, eventually coming back in another burst. From what has been observed so far, this seems to be a recurring pattern of behaviour that is mixed with infrequent, longer interaction sequences.

A logical next step now is to see if the same behaviour is representative across the whole user population. The interaction bursts and the behaviour variety in the form of longer and shorter connected sequences are mainly dependent on the construct of lead-ins. Therefore, the optimal way to see if this type of behaviour is representative is to analyse these sequences in an aggregated way. To do so, the mechanical session-type of visits is best suited since it aims to aggregate such sequences into individual sessions. The only difference is the click path analysis, which may interfere with the order of these types of sequences. Therefore, the visits can be used as the ground-truth for further analysis. This circumstance is then used to compare and analyse all other session approaches and their assumptions afterwards to see how they can reflect behaviour and look at their impact on the numbers.

The next section first analyses the visits and assesses their potential to reflect the behaviour presented in this section. The following sections are then focused on all other session approaches to see how their assumptions hold up to the suppositions made in this section and to analyse the impact of the differing methods on the respective measures.

5.2 Analysis of Session-Identification Approaches

The intention of this section is to quantify the impact of session modelling on several evaluation measures related to system performance and user behaviour. The first section deals with the mechanical session approaches and takes a look at the structural visit approach, temporal inactivity and fixed time ranges. The second section evaluates the logical session approaches, starting with an analysis of the baseline approaches using lexical matching and followed by analysing the BM25 approaches using shared vocabulary. The approaches using different variations of word2vec in the form of user traffic to vectors are analysed. The third section repeats that analysis with the combined approaches. Finally, a comparative discussion looks at the differences between all the approaches in a summarized form. The structure of the analysis is simple. The following measures are calculated at **session_id** level:

- System-related measures (number of sessions, conversion rate, lead-ins, bounce rate)
- User-related measures (sessions per user, time spent, number of visited categories, products and issued queries, interactions per session, typical page sequences)
- Approach-related measures (number of root categories, number of topics, number of breaks)

By looking at these indicators, the differences between the variants are discussed and put in context to other approaches.

5.2.1 Mechanical Sessions

This section evaluates the mechanical session approaches and their impact on the previously explained measures. The first section is about the visits. The second section is about the sessions identified by a maximum session length. The final section explains sessions identified by temporal inactivity: the first part with a fixed threshold, the second part with a dynamic threshold. In the following considerations, visits is used synonymously when analysing the session approach based on the **visit_id**. Approaches using an inactivity threshold are also called timeout sessions. The method using a fixed maximum time is also called a fixed length session.

5.2.1.1 Structural Sessions – Visits

The visits are the most fundamental session approach. As defined in Section 4.1, visits are technically a different concept when compared to sessions. They have no underlying assumption since they simply replicate user behaviour (as far as the data allows). By performing a click path analysis, the visits try to reconstruct the individual interaction sequences a user performs on the system with an entry point and an explicit last interaction. There is no assumption about any information need, just the plain path representation. By relying on a click path analysis, this approach is directly dependent on the quality of the dataset; if there are untracked pages or the content of the tracked pages is incomplete or patchy, there will be session breaks and incomplete sequences. This may lead to misleading interpretations of user behaviour, because visits and, therefore, sequences are cut short or are incorrectly connected.

Overall, there are 513,007,900 visits. The average visit has a length of around 2.83 interactions with a standard deviation of 2.92. The median is even more meaningful with a value of two. The average number of visits per **user_id** is 6.55 with a standard deviation of 19.21 and a median of three. The maximum number of visits is 7,066. When using the number of visits as a dimension, this measure has a cardinality of 1,917. In Figure 5.14, the

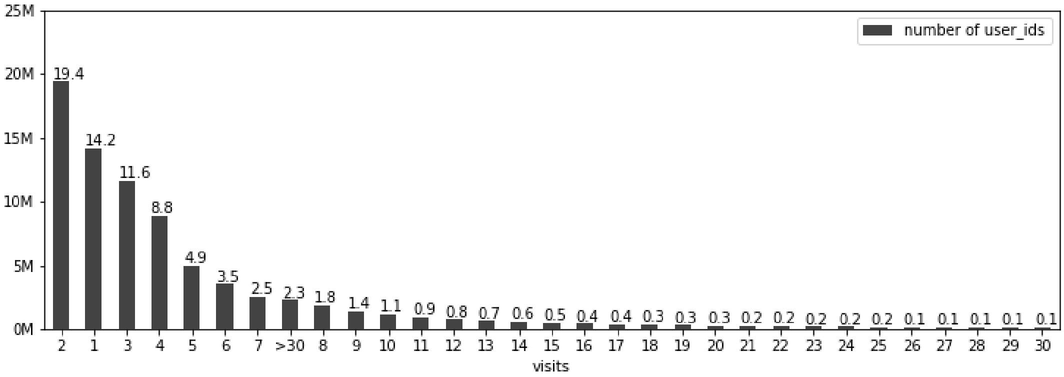


Figure 5.14: Number of users per visits as a dimension.

distribution of unique **user_ids** per **visit_id** as a dimension is shown. The vast majority of **user_ids** (88.39%) have less than or equal to 10 visits. The highest share with around 42% of all **user_ids** make only one or two visits. These users amount to only roughly 13.2% of all interactions though. The **user_ids** with less than 10 visits make up 43.32% of all interactions while the remaining 11.61% of **user_ids** with more than 10 visits makes 56.68% of all interactions. Logically, the majority of the first two interaction buckets make less than 10 visits, whereas the other buckets make more than 10 visits. The evidence is not clear, but there is an indicator to support the supposition that user behaviour stays somewhat similar with rising engagement with the system; that is, the more interactions a user has, the more visits s/he will make, but the structure of those visits will remain similar to an extended degree.

Table 5.2 shows the same metrics as the previous section, but calculated again on a **visit_id** basis. **visit_id** basis. The same interaction buckets are retained to see if a divergence on a sequence basis can be identified and to see if there is a difference in user behaviour between the buckets. Additionally, the average time spent on the system is calculated for every visit session.

	Total		≤ 10		>10,≤ 30		>30,≤ 100		>100,≤ 500		>500	
	AM	SD	AM	SD	AM	SD	AM	SD	AM	SD	AM	SD
interactions	2.47	3.87	1.91	1.45	2.47	2.75	2.57	3.6	2.77	4.39	3.48	8.71
interaction days	1.02	0.36	1.01	0.11	1.02	0.19	1.03	0.3	1.04	0.51	1.06	0.79
root categories	1.18	0.44	1.13	0.37	1.14	0.4	1.16	0.43	1.23	0.5	1.37	0.61
categories	1.27	0.7	1.2	0.5	1.22	0.61	1.25	0.66	1.34	0.78	1.56	1.18
queries	0.36	0.91	0.36	0.58	0.31	0.68	0.33	0.76	0.37	0.93	0.52	2.06
products	0.89	1.08	0.75	0.69	0.9	0.91	0.93	1.0	0.97	1.15	1.11	2.13
time btw. ints.	83.5	2,109.33	51.87	1,566.64	71.8	1,908.66	97.65	2,357.51	105.29	2,388.07	119.67	2,578.39
time in session	193.52	4,708.25	69.22	2,496.41	147.04	3,956.46	225.3	5,127.87	289.78	5,726.09	446.73	7,769.17
conversion rate	48.72%		34.06%		53.01%		53.87%		54.92%		58.64%	
visit share	100%		27.91%		22.91%		21.86%		19.61%		7.71%	

Table 5.2: Summary statistics for visits per interaction bucket. Measures are calculated as an average of the distinct count of the respective measure per individual visit_id. Abbreviations: AM = Arithmetic Mean, SD = Standard Deviation.

The table paints a clear picture of how the system is visited. Overall, the differences between the interaction buckets are relatively minor but definitely noticeable. The totals are as expected. The average of 2.47 interactions (along with the standard deviation of 3.87 interactions) matches the examples shown in the previous section. Considering the visited categories, products and issued queries, the assumptions made seem to be correct as well: on average, one to two categories are worked on in a single visit with a tendency towards working on exactly one category. This corresponds with the small number of interactions here. The time between interactions and the overall time on site is surprisingly high though; this may be caused by artefacts in the data (i.e. tracking issues, methodical errors and power users disturbing the distribution). The high-standard deviation seems to indicate the assertions about possible artefacts as well.

The differences between interaction buckets are not too great. The average number of interactions per visit grows in line with the total number of interactions a user makes. The rate the average rises is another indicator that a high volume of visits still remains relatively similar no matter the bucket; the rise in the standard deviation also supports the previous assumption that there is a mixture of behaviour with a rising number of longer visits alongside the typical short-sequence visits. Looking at the numbers related to the content again, the average does not change significantly across the buckets where slightly higher values represent the more engaged users. There is no real difference between the interaction buckets and the total values; even the standard deviations are somewhat alike. This is another indicator of similar behaviour; the average visit will only deal with one or two topics with a tendency towards one topic; an exception are the power users according to the standard deviations.

The time between interactions is not very surprising either. On average, the inter-interaction time for a visit is around 83 minutes. There is a difference between the buckets, which may be caused by the mixture of behaviours in the higher interaction buckets; whilst the lower interaction buckets almost exclusively make very short visits that tend to come in the previously mentioned bursts, the higher variants show a longer time is spent between interactions. The inter-interaction time is generally relatively high but is most likely disturbed by longer sequences. There is not much to learn from these numbers; one assumption could be that the more interactions a user makes, the more likely it is that they will spend more time in one session, thereby generating higher inter-interaction times (and a larger time in session as can be seen in the table).

The last rows of the table show the actual share of visits per interaction bucket and, related to that, the conversion rate calculated on a visit level and aggregated per interaction bucket. The share of visits per interaction bucket is close to the distribution of interactions per interaction bucket. The number of interactions per visit foreshadows the share of visits per bucket. The more interactions a **user_id** has, the more likely it is that the visits will get longer. The conversion rates are as expected as well. The higher buckets deliver the highest conversion rates in terms of making an average number of visits with a relatively high and rising number of *leadouts*. The lowest bucket simply does not make that many *leadouts* compared to the quantity of sessions. In total, a conversion rate of 48.72% seems somewhat reasonable considering the type of system.

Interaction Bucket	Sequences	Totals
All Users	li_oop (26.06%) — li_q (8.54%) — li_pc (5.94%) — li_oop, lo (5.18%) — li_hp (4.0%)	49.72%
$\leq$ 10	li_oop (28.5%) — li_q (11.64%) — li_pc (7.05%) — li_oop, lo (5.09%) — li_q, oop (3.65%)	55.93%
>10,$\leq$ 30	li_oop (25.38%) — li_q (8.5%) — li_pc (7.08%) — li_oop, lo (5.34%) — li_oop, oop (3.79%)	50.09%
>30, $\leq$ 100	li_oop (26.31%) — li_q (8.24%) — li_pc (5.96%) — li_oop, lo (5.6%) — li_oop, oop (3.62%)	49.73%
>100, $\leq$ 500	li_oop (25.59%) — li_q (6.56%) — li_hp (5.55%) — li_oop, lo (5.29%) — li_pc (4.34%)	47.33%
>500	li_oop (19.69%) — li_hp (11.38%) — li_hp, oop (4.47%) — oop (3.99%) — li_oop, lo (3.56%)	43.09%

Table 5.3: Top five sequences for visits per interaction bucket.

Table 5.3 lists the top five visit sequences by number of appearances in the data, shown per interaction bucket. The share of these sequences on all visits is also shown as a percentage. The bottom row shows the total share of sequences in all visit sequences in the respective bucket. As can be seen at a glance, the top sequences are all very similar.

The top sequence for all buckets is exactly the same: a single click lead-in on a product page. With around 25% for all buckets except the power user bucket, this is somewhat clear evidence of how the majority of users visit the system. All other sequences are also very short visits, the highest counting three clicks among the top five list. The table clarifies that the general behaviour is very similar. With more and more interactions, the users will mix-in longer sequences. This can be extrapolated from three things: first, the buckets with more interactions have a growing number of slightly longer sequences in the top 10; and two, the total share of the top five sequences decreases with the growing number of interactions; and three, since the average number of visits rise with the growth in interactions, the additional interactions must belong exactly with these longer sequences. Especially the decrease of the total share indicates a greater variety of different sequences.

The share of interactions on different sequences between the interaction buckets is also interesting. Visits with a single click on the homepage receive a higher share in the higher interaction buckets. This is an indicator of more engagement; these users have to perform a conscious act to visit the system as the homepage is only reachable when directly searching for the system (or typing out the address). When looking at the overall numbers of sequences, it is still very surprising to only see sequences with a maximum number of three clicks. This is a profound indicator of system usage: users will more often use the system for a quick search on a topic. Actual longer visits are relatively rare, 51.5% of all visits have only one interaction.

To underline this point, interleaving behaviour within the visits should be analysed. Interleaving behaviour would indicate multitasking; in the case of visits, this would mean that the user travels multiple click paths at once. Click path analysis is ideal for this, since it actually shows if a user works in multiple sequences. In total, only 1.6% of all visits show interleaving properties. This means that the vast majority of these sessions are straightforward sequences, whereas around 8 million show interleaving paths. That is both low and very surprising, but a clear sign of the behaviour type described. The average number of potential topics as defined in Section 4.5) is actually very similar to the number of categories and root categories per visit as seen in Table 5.2: 1.5 potential topics on average per visit. This is intuitively understandable as the topics are closely connected to the category tree. In any case, the low number also follows the assumptions about user behaviour; users will visit the system in short sequences and will more than likely work only on one topic in one subsequent sequence with no interruption.

This is an interesting conclusion to note for all following mechanical session approaches. If in general user behaviour rarely shows interleaving behaviour and a focus on short sequences with a limited number of topics, the basic assumption about mechanical sessions is clearly fulfilled. Therefore, some variant of timeout sessions might be capable of replicating user behaviour just as the visits do. The visits as a session approach seem to capture user behaviour quite well overall, although no real estimation can be given about how grave an impact the mentioned bugs will actually have. The biggest downside of using visits alongside these potential errors is the computational complexity, especially in a live environment. Keeping track of all session states for all **user_ids** is potentially a very speed-limiting factor that can be seen as a big disadvantage. From a qualitative perspective (and assuming that the data quality is good enough), visits may be the closest approach to actually replicate user behaviour on the system.

The data quality is actually the key here when talking about the qualitative perspective. If the quality is good enough, visits are a great way to observe user behaviour but they are completely dependent on the data. This is clearly a disadvantage for this type of session identification because the potential for errors and falsely constructed visits is great. There are many examples of errors in the quality of the data that could be responsible for incorrect session identification. Most prominently, an example may lie in the lack of an **http_referer** or, more precisely, a loss of valid interactions. In a lot of cases, **user_ids** visit the system with no referer at all – due to browser settings or via direct access by bookmark or typing the URL in the address line. The algorithm cannot differentiate between these cases as there is no information present. Another error variant is a new visit from an internal page that was not seen in the user history before. This can mean either that the **cookie_value** or the **user_id** has been incorrectly set or, more likely, that some events have been filtered or removed in the tracking due of certain conditions – a prime example of this being an invalid HTTP status. Directly related to this is the existence of parameters that are only added to the **url** in fully loaded interactions; sometimes, these parameters are not visible in the **http_referer**, making it impossible to connect these events. The same is true for **user_ids** manually manipulating **url** parameters or going back and forth between pages with **url** parameters. If the user visits a page with external parameters multiple times, there is no way to decide if these are new visits and to which click path they actually belong.

The extent of these errors is for the most part not fully measurable. Session breaks caused by mismatches between **url**- and **http_ referer** are especially difficult to detect without digging very deep. A rough estimate can be made by counting the number of visits starting directly from an idealo-page. This affects around 57 millon visits – roughly 11% of the overall total. Additionally, there are 135m visits starting with an empty **http_referer**, accounting for 26% of the overall amount. These are only rough estimates, and especially the empty **http_referers** need not necessarily be incorrect session breaks; 37% (and probably even more) potentially incorrect session breaks does seem to be a high number though.

Other approaches do not have these problems because they rely on different fields; the following sections will show how they differ from a numbers' perspective.

5.2.1.2 Temporal Sessions – Fixed Length

The basic assumption for mechanical sessions in general and temporal sessions specifically is that users will spend a certain amount of time on a single subject before moving onto the next subject. Fixed length sessions (also referred to using the identifier **tf** (temporal fixed)) are probably the oldest and most basic variant of sessions. This approach takes the assumption very literally, trying to estimate the time a user will most likely work on a given subject. After the time is up, a hard boundary is set and a new session begins.

With knowledge of what the visits look like and how very close to the base assumption of the mechanical sessions they are, analysis of the fixed length sessions should reveal some interesting insights. The difference between the visits and the mechanical sessions assumption is that visits may be in close temporal proximity and that it's usually a lead-in to the system that starts a visit. The fixed length session may be able to capture that behaviour and these sequences under the assumption that there is a typical temporal time frame for such sequences to occur.

	#Sessions	CV-R	B-R
tf5	426,196,079	58.64%	37.49%
tf10	379,381,854	65.87%	36.07%
tf15	359,698,513	69.48%	35.37%
tf20	348,197,625	71.77%	34.9%
tf30	334,675,756	74.67%	34.27%
tf45	323,595,977	77.23%	33.64%
tf60	316,804,935	78.89%	33.2%
tf90	308,159,712	81.1%	32.53%
tf120	302,448,723	82.63%	32.04%
tf180	294,520,625	84.86%	31.32%
tf360	281,207,349	88.87%	30.08%
tf720	267,565,842	93.4%	28.61%
tf1440	246,739,006	101.29%	26.33%
tfd	267,188,092	93.54%	28.8%

Table 5.4: System measures for tf sessions. Abbreviations: CV-R = Conversion Rate; B-R = Bounce Rate.

The descriptive statistics in Table 5.4 describe the impact on the system of the tested fixed length variants. These are calculated by comparing the overall numbers on an approach level; the measures are not calculated on a **user_id** basis, but simply by comparing the sum of the respective measure with the calculated number of sessions. The number of sessions, the conversion rate and the bounce rate are shown as they are essential for measuring a system's performance from a business perspective. perspective.

The results are not surprising: The number of sessions decreases with a longer fixed time length; this was expected and is only logical. The number of sessions does not develop in the same way the time parameter does though: where the allowed time period literally doubles from five to 10 minutes between **tf5** and **tf10**, the number of sessions only decreases by roughly 11%. This is even more interesting when comparing **tf15** to **tf30** or **tf30** to **tf60**, with a decrease of around 7% and 5.3% respectively. The difference gets even smaller with a longer fixed length. This is an indicator of somewhat uniform behaviour among the majority of users; a fixed parameter for sessions would most likely be able to capture the behaviour for a large part of the user population.

While the *leadouts* are a fixed entity (there is an absolute number of *leadouts* which does not change with the number of sessions), the conversion rate is directly dependent on the absolute number of sessions. With this knowledge in mind, the numbers are very

	≤ 10	>10,≤ 30	>30, ≤ 100	>100, ≤ 500	>500
tf5	27.16%	22.1%	21.85%	20.2%	8.7%
tf10	27.99%	21.92%	21.73%	19.98%	8.38%
tf15	28.5%	21.86%	21.67%	19.82%	8.16%
tf20	28.84%	21.85%	21.62%	19.7%	7.99%
tf30	29.31%	21.86%	21.57%	19.52%	7.75%
tf45	29.74%	21.9%	21.53%	19.33%	7.49%
tf60	30.04%	21.95%	21.51%	19.2%	7.3%
tf90	30.44%	22.05%	21.49%	19.0%	7.02%
tf120	30.73%	22.13%	21.48%	18.84%	6.82%
tf180	31.17%	22.25%	21.47%	18.6%	6.51%
tf360	32.0%	22.52%	21.44%	18.1%	5.94%
tf720	32.99%	22.86%	21.38%	17.47%	5.3%
tf1440	34.58%	23.27%	21.11%	16.43%	4.6%
tfd	33.17%	22.95%	21.4%	17.35%	5.13%

Table 5.5: Share of tf sessions per interaction bucket.

reasonable. As expected and hardly surprising, the rate decreases with the number of sessions. Another consideration is the fact that the conversion rate is directly dependent on the number of sessions, which is again apparently directly dependent on the session approach. This will be more interesting to explore in comparison with the other approaches.

The bounce rate is far more interesting than the conversion rate. Bearing in mind that the number of sessions with only one interaction is directly dependent on the session-identification algorithm, the bounce rate provides a clear indication of how well this approach captures user behaviour. The differences directly hint at how the maximum length fits the sequences on the system: these range from 37.49% sessions with only one interaction for **tf5** to 26.33% sessions for **tf1440**. The rate for the **tfd** sessions, grouping all events of a user per day into one session is at 28.8%, which implies that there could be a viable fixed length. The minimal difference between **tf90** and **tf120** as well as **tf720** and **tfd** in terms of the total number of sessions and bounce rate is especially interesting; essentially, there is only a very minor difference here especially for the latter. Theoretically, this indicates that most users will perform all interactions within a 720-minute time frame. The relatively steep decline of total sessions from **tf180** to **tf360** also indicates a specific length that captures the majority of user interactions. All interactions could probably be grouped into sessions with a time parameter between three and six hours without losing that many sessions by increasing the length.

Looking at the numbers of sessions per interaction bucket in Table 5.5, there are only small differences. There are no real surprises here; the distribution is relatively uniform. One of the bigger differences can be seen for the **tf1440** session approach regarding the users with more than 500 overall interactions. Logically, these will have a smaller share of sessions with a maximum length of 1,440 minutes, due to the huge number of interactions over the course of the year. Likewise, the session approaches with a shorter length logically have a lower share in the buckets with less overall interactions.

Having gained this impression from the overall numbers, now the focus shifts to the different measures on the **user_id** and **session_id** levels. The goal is to understand the nuances of the two session approaches and to look at whether the differences between users affect the the overall numbers regarding each session approach. Every measure is calculated

	∅Sessions		∅Interactions		CV-R		B-R		Lead-ins	
	AM	SD	AM	SD	AM	SD	AM	SD	AM	SD
tf5	5.44	17.54	3.16	2.05	56.75%	76.92%	33.14%	32.85%	1.33	0.8
tf10	4.84	15.18	3.69	2.74	68.73%	107.66%	29.4%	32.64%	1.5	1.02
tf15	4.59	14.06	3.95	3.12	74.86%	119.58%	27.78%	32.41%	1.58	1.08
tf20	4.44	13.39	4.11	3.39	78.74%	127.12%	26.83%	32.22%	1.63	1.12
tf30	4.27	12.55	4.32	3.75	83.51%	139.4%	25.7%	31.95%	1.69	1.2
tf45	4.13	11.77	4.49	4.07	87.44%	147.8%	24.76%	31.68%	1.74	1.24
tf60	4.04	11.28	4.59	4.28	89.76%	152.96%	24.19%	31.48%	1.78	1.27
tf90	3.93	10.6	4.71	4.53	92.49%	159.1%	23.44%	31.2%	1.81	1.3
tf120	3.86	10.16	4.79	4.69	94.13%	162.73%	22.95%	31.0%	1.84	1.32
tf180	3.76	9.53	4.89	4.88	96.19%	167.13%	22.28%	30.7%	1.87	1.35
tf360	3.59	8.49	5.04	5.18	99.37%	173.47%	21.21%	30.18%	1.93	1.4
tf720	3.41	7.44	5.18	5.42	102.09%	178.21%	20.11%	29.61%	1.98	1.45
tf1440	3.15	6.22	5.41	5.7	106.75%	184.84%	18.32%	28.6%	2.07	1.53
tfd	3.41	7.27	5.17	5.42	101.82%	177.79%	20.33%	29.74%	1.97	1.46

Table 5.6: User measures for tf sessions regarding system usage. Abbreviations: CV-R = Conversion Rate; B-R = Bounce Rate; AM = Arithmetic Mean; SD = Standard Deviation.

on a **user_id** and **session_id** level and then aggregated again by session approach. Table 5.6 shows the statistics.

The averages between the different approaches are quite different. As expected, the average number of sessions per user decreases in line with the chosen maximum length. Although the differences between the variants are quite large, they are not as big as one might expect considering that the shortest maximum length is just five minutes and the longest is a whole day. The variance ranges from 5.44 sessions on average for **tf5** up to 3.15 sessions on average for **tf1440**. There is only a small difference between **tf720** and **tfd**. This is intuitively understandable: the majority of interactions are likely to happen within the bounds of a day with only the occasional sequence of interactions spanning different days, as reflected in the fewer sessions of **tf1440** compared to **tfd**.

The average interactions per session behave in the same way, which is logical. The longer the chosen maximum-length parameter, the greater the number of interactions on average per session, suggesting that the interactions are at least happening somewhat regularly. The distribution is similar to the number of sessions per user. The variance ranges from 3.16 to 5.17. The observed similarities between the approaches persist. Interestingly, the conversion rate is very similar compared to the conversion rate across all the approach data. The same is true for the bounce rate, although the divergences are more obvious here. The **tfd** sessions are a good indicator for further reasoning since they combine all sessions of a single calendar day into one session. When looking at the bounce rate here, it can be safely stated that 20.33% of the calculated visits have exactly one interaction and these are the only interactions of that user on this respective day.

The lead-ins per session are all somewhat similar with a variance ranging from 1.33 for **tf5** to 2.07 for **tf1440**. The differences indicate mixed behaviour; often, sequences of interactions on the site are relatively short and start routinely with a new lead-in. This is even reflected in the difference between **tf5** and **tf10**; assuming most of these sessions start with the same event (at least for users starting their sessions on different days), the difference in lead-ins is relatively high. With this knowledge, one could argue that the fixed length sessions may not be able to completely replicate the user behaviour as

	⌀Root Categories		⌀Categories		⌀Products		⌀Queries		⌀Topics	
	AM	SD	AM	SD	AM	SD	AM	SD	AM	SD
tf5	1.26	0.46	1.44	0.71	1.05	0.83	0.55	0.82	1.27	0.51
tf10	1.29	0.5	1.5	0.79	1.15	0.98	0.6	0.9	1.31	0.56
tf15	1.31	0.51	1.53	0.83	1.2	1.07	0.62	0.93	1.32	0.58
tf20	1.31	0.52	1.55	0.85	1.24	1.13	0.64	0.96	1.33	0.6
tf30	1.33	0.54	1.57	0.88	1.28	1.2	0.66	0.99	1.35	0.62
tf45	1.34	0.55	1.59	0.91	1.32	1.27	0.68	1.02	1.36	0.63
tf60	1.34	0.55	1.6	0.93	1.34	1.32	0.69	1.05	1.37	0.64
tf90	1.35	0.56	1.62	0.95	1.36	1.37	0.7	1.08	1.38	0.66
tf120	1.35	0.57	1.63	0.96	1.38	1.41	0.71	1.11	1.39	0.66
tf180	1.36	0.57	1.64	0.98	1.4	1.46	0.72	1.14	1.39	0.68
tf360	1.37	0.59	1.67	1.01	1.44	1.53	0.73	1.2	1.41	0.69
tf720	1.38	0.59	1.69	1.04	1.46	1.6	0.75	1.26	1.42	0.71
tf1440	1.4	0.61	1.72	1.07	1.51	1.66	0.77	1.3	1.45	0.73
tfd	1.38	0.59	1.68	1.04	1.46	1.6	0.74	1.26	1.42	0.71

Table 5.7: User measures for tf sessions regarding visited content. Abbreviations: AM = Arithmetic Mean; SD = Standard Deviation.

shown before. Still, average lead-ins per session is surprisingly low (also considering the low standard deviation), so they may be good enough to show the movements of a user on the page.

Looking at the content measures in Table 5.7, there are only small differences between the different approaches. This comes as a bit of a surprise, but it is also understandable considering the type of system and how the users are searching. There are almost no differences in the average number of distinct root and product categories visited (with a logical tendency towards a higher number for the longer maximum-length variants). The same is true for products, queries and the number of topics. The most interesting takeaway here is that these values are similar for the **tfd** sessions too, which consolidate all interactions per interaction day. This means that the general behaviour of users here is actually relatively well depicted; mostly, it is only a small number of content-related measures that are visited per interaction day. The small differences compared to the shorter maximum-length sessions are relatively surprising nonetheless.

	⌀Time in session		⌀Inter-Interactiontime		⌀Interaction days	
	AM	SD	AM	SD	AM	SD
tf5	0.85	0.95	0.67	0.56	1.0	0.01
tf10	1.71	2.07	0.94	1.06	1.0	0.02
tf15	2.38	2.99	1.16	1.5	1.0	0.02
tf20	2.93	3.84	1.34	1.89	1.0	0.02
tf30	3.86	5.39	1.65	2.63	1.0	0.02
tf45	5.0	7.47	2.05	3.65	1.0	0.03
tf60	5.96	9.4	2.39	4.63	1.0	0.03
tf90	7.64	13.02	3.04	6.6	1.0	0.03
tf120	9.16	16.51	3.65	8.58	1.0	0.03
tf180	12.07	23.52	4.86	12.57	1.0	0.04
tf360	20.16	44.01	8.24	24.13	1.0	0.05
tf720	35.71	84.46	15.2	48.91	1.01	0.09
tf1440	97.32	224.27	40.41	124.27	1.06	0.19
tfd	35.33	88.27	14.66	49.35	1.0	0.0

Table 5.8: User measures for tf sessions regarding time spent. Abbreviations: AM = Arithmetic Mean; SD = Standard Deviation.

As with the content measures, the time-related measures shown in Table 5.8 are not that surprising. There are noticeable differences between the variants, but they are very reasonable considering that the maximum length has a direct impact on any time-related measure. Interestingly, the average time in session rises, but not in the same way the maximum-length parameter rises, indicating again that there is indeed a maximum length of time that may capture the majority of related sequences. The same is true for the inter-interaction time. Here, the most interesting finding relates to the inter-interaction time, which stays at a comparatively low level; up to **tf360**, it still averages at below 10 minutes between all interactions, but the interactions per session increases for these same sessions. Meanwhile, the interaction days logically remain very much the same with only a small divergence for the higher-length sessions.

Finally, Table A3 shows the top five sequences. Here, there is no real noticeable difference between the different approaches. As reflected in the table, the sequences of all session approaches remain the same for all short sessions occurring on one day. A session starting at 8 a.m. with two interactions and an inter-interaction time lower than five minutes, will be the same no matter the maximum-allowed length set by the session-identification algorithm. Taking into consideration that by and large all the sessions apparently mirror this (around between 30% and 40% according to the bounce rate), the sequences, their shares and the total chosen for the top five are reasonable. An interesting observation is that the order of sequences changes from **tf45** to **tf60**, with two lead-in clicks now on position five instead of on a single product-page click.

This concludes the description of the maximum-length sessions. It is evident that there are noticeable differences particularly in terms of sessions per user and interactions per session across all approaches. The different levels of increase (or decrease, for that matter) between the different approaches' maximum length indicates that there is indeed a maximum length that captures temporally close interactions across the user base. In view of the fact that the average visited categories and topics are also comparably low, the majority of these sessions might also, therefore, be related to the same information need. The next section will show how temporal inactivity instead of maximum length defines sessions.

5.2.1.3 Temporal Sessions – Inactivity Timeout

The purpose of inactivity timeout sessions is not so much to try to find the maximum length of a typical session, but rather to intuit a valid inactivity time period that represents the user's inactivity between two different sessions. Since estimation of the maximum length of time has so far not been that effective, the goal here is to find a common period of time between two different topics that could be representative of the inactivity period used to identify session boundaries. Assuming that such a timeout can be found and actually represent user behaviour accordingly, the inactivity timeout sessions may be even better at replicating user behaviour in terms of visits, as time periods can be a somewhat variable means of working this out. This section describes all the temporal inactivity approaches to see how the different timeouts change these figures.

This dissertation tests two types of temporal inactivity approaches: the first of these are the fixed types of inactivity threshold; if the time between two interactions is longer than a fixed specified threshold, a new session begins. The second type are variants with a dynamic threshold that changes depending on certain conditions like the visited page type or the visited category. In total, 22 different session approaches are analysed: 13

with a fixed inactivity timeout (beginning with the prefix **ti**) and nine with a dynamically calculated inactivity timeout (beginning with the prefix **td**).

Fixed inactivity timeout

The fixed inactivity timeout sessions assume that there is a global timeout capable of reflecting the activity behaviour of all users. The assumption is that the inactivity time between dealing with different topics or using the system in different sessions is the same for the whole user population. The tested values range from five minutes of inactivity to a complete day of inactivity with 1,440 minutes. The industry standard of 30 minutes inactivity is included here as well as its spiritual predecessor of 26 minutes (originally 25.5 minutes) introduced by Catledge and Pitkow in 1995 [43].

	#Sessions	CV-R	B-R
ti5	388,179,295	64.38%	38.63%
ti10	358,125,465	69.78%	36.8%
ti15	344,444,702	72.56%	35.9%
ti26	329,331,338	75.89%	34.83%
ti30	325,924,065	76.68%	34.57%
ti45	317,139,738	78.8%	33.86%
ti60	311,459,298	80.24%	33.36%
ti90	303,731,866	82.28%	32.66%
ti120	298,319,596	83.78%	32.16%
ti180	290,545,289	86.02%	31.44%
ti360	277,612,404	90.02%	30.16%
ti720	263,709,857	94.77%	28.73%
ti1440	235,395,534	106.17%	26.83%

Table 5.9: System measures for ti sessions. Abbreviations: CV-R = Conversion Rate; B-R = Bounce Rate.

Table 5.9 gives an overview of the global system measures for the fixed-inactivity timeout sessions. There are tremendous differences in the number of sessions between the different timeouts. The variance ranges from 388m five-minute timeout sessions to only 235m **ti1440** sessions, that is, those with a 1,440-minute inactivity timeout. There is a difference of roughly 30m sessions between the five-minute inactivity timeout and the 10-minute timeout. Interestingly, the well-known 30-minute inactivity timeout identifies 325m sessions, whereas the **ti26** as its originator results in 329m sessions. The differences are quite large and do not seem to follow an easily understandable pattern. The number of sessions decreases drastically the longer the allowed inactivity timeout is, which is very reasonable. The conversion rate follows the same behaviour.

The bounce rate is actually very similar compared to the maximum-length sessions, which is quite surprising since the approaches work rather differently. Checking the overlap here could be a good way to find where the remaining differences come from; this would help in identifying a suitable maximum length and a global inactivity timeout if needed. In any case, as for the **tf** sessions, roughly 30% of the identified sessions are bounced sessions with only one interaction per session.

The differences between the **ti** and **tf** sessions are astonishingly small. Starting from the 30-minute temporal threshold, both approach types identify a very similar number of sessions when comparing the respective temporal boundary. For example, **ti30** and **tf30** as well as **ti720** and **tf720** show almost the same number of sessions. Analysing the data, these approaches have a high overlap, almost identifying the exact same sessions. As an example, 318m of the 325m **ti30** sessions are exactly the same session; in other words, 95% of all interactions in the data belong to the same **ti30** and **tf30** session. Similar numbers can be observed for the other temporal boundaries. This is a highly interesting observation. To stick with the well-known 30-minute boundary, what this observation points to is that only 5% of the interactions in this data take place more than 30 minutes after the first interaction of a session.

	$\leq$ 10	>10,$\leq$ 30	>30, $\leq$ 100	>100, $\leq$ 500	>500
ti5	28.46%	21.8%	21.54%	19.88%	8.31%
ti10	29.01%	21.79%	21.51%	19.69%	8.0%
ti15	29.33%	21.82%	21.5%	19.56%	7.79%
ti26	29.76%	21.88%	21.49%	19.37%	7.51%
ti30	29.88%	21.9%	21.49%	19.31%	7.43%
ti45	30.2%	21.97%	21.48%	19.16%	7.19%
ti60	30.44%	22.03%	21.48%	19.04%	7.01%
ti90	30.8%	22.14%	21.49%	18.85%	6.72%
ti120	31.08%	22.23%	21.49%	18.69%	6.51%
ti180	31.53%	22.38%	21.49%	18.42%	6.17%
ti360	32.36%	22.67%	21.47%	17.9%	5.6%
ti720	33.41%	23.04%	21.4%	17.2%	4.94%
ti1440	36.1%	23.91%	21.12%	15.41%	3.47%

Table 5.10: Share of ti sessions per interaction bucket.

Looking at Table 5.10 there are no real differences between the distribution of the session approaches in the interaction buckets. Once again, the distributions are very similar to the maximum length sessions: session approaches with a higher timeout have a lower share in the higher interaction buckets and vice versa. Again, this is very reasonable, since these users are likely to visit the page daily or at least on a very regular basis, making a 1,440-minute break between interactions a rarer observation because of the frequency of interactions. It is quite interesting to see that the majority of differences can be found in the lowest and the two power user buckets; the users in-between with more than 10 and less or equal to 100 interactions share roughly the same number of sessions across all timeout variants. This may signal uniform behaviour in these buckets with regards to the inter-interaction time and the general system usage.

	∅Sessions		∅Interactions		CV-R		B-R		Lead-ins	
	AM	SD	AM	SD	AM	SD	AM	SD	AM	SD
ti5	4.95	15.56	3.72	3.36	69.13%	120.78%	31.99%	33.33%	1.47	1.1
ti10	4.57	13.88	4.09	3.86	78.16%	137.03%	28.77%	32.84%	1.6	1.18
ti15	4.4	13.03	4.27	4.12	82.49%	144.87%	27.34%	32.52%	1.66	1.22
ti26	4.2	12.08	4.48	4.42	87.27%	153.59%	25.79%	32.1%	1.72	1.27
ti30	4.16	11.85	4.52	4.49	88.31%	155.45%	25.45%	31.99%	1.74	1.28
ti45	4.05	11.16	4.64	4.67	90.9%	160.11%	24.59%	31.7%	1.78	1.3
ti60	3.97	10.73	4.71	4.79	92.46%	162.94%	24.05%	31.5%	1.8	1.32
ti90	3.88	10.1	4.8	4.95	94.45%	166.55%	23.34%	31.21%	1.83	1.35
ti120	3.81	9.66	4.87	5.07	95.76%	168.88%	22.87%	31.0%	1.86	1.42
ti180	3.71	9.04	4.95	5.21	97.55%	172.02%	22.21%	30.7%	1.89	1.45
ti360	3.54	8.06	5.09	5.4	100.26%	176.44%	21.15%	30.18%	1.94	1.51
ti720	3.37	7.04	5.23	5.75	102.92%	181.51%	20.05%	29.6%	1.99	1.57
ti1440	3.0	5.25	5.59	7.01	109.96%	199.74%	18.2%	28.6%	2.13	1.95

Table 5.11: User measures for ti sessions regarding system usage. Abbreviations: CV-R = Conversion Rate; B-R = Bounce Rate; AM = Arithmetic Mean; SD = Standard Deviation.

Looking at the numbers in Table 5.11, the variances among the average sessions per user are smaller than for the **tf** sessions seen in Table 5.6. However, bearing in mind the identification method, this is reasonable; looking at arbitrary inactivity time frames between events to connect interactions should be more likely to result in a connected session than connecting interactions within an arbitrary maximum length. On average, there are 4.95 sessions per user for a five-minute timeout, decreasing to 3.0 sessions per user for a 1,440-minute timeout. The average interactions seem to follow the same pattern, with 3.72 interactions on average increasing to an average of 5.59 interactions.

More or less replicating the global measure, the conversion rate offers no surprises. The bounce rate differs more widely: for the five-minute timeout it remains at around 30%,

	∅Root Categories		∅Categories		∅Products		∅Queries		∅Topics	
	AM	SD	AM	SD	AM	SD	AM	SD	AM	SD
ti5	1.29	0.5	1.49	0.81	1.15	1.07	0.6	0.93	1.3	0.57
ti10	1.31	0.52	1.54	0.86	1.23	1.2	0.64	0.99	1.33	0.6
ti15	1.32	0.53	1.56	0.89	1.26	1.26	0.65	1.02	1.34	0.61
ti26	1.33	0.55	1.58	0.92	1.31	1.33	0.67	1.07	1.36	0.63
ti30	1.33	0.55	1.59	0.93	1.32	1.35	0.68	1.08	1.36	0.64
ti45	1.34	0.56	1.61	0.95	1.35	1.4	0.69	1.12	1.37	0.65
ti60	1.35	0.56	1.62	0.96	1.36	1.43	0.7	1.14	1.38	0.66
ti90	1.35	0.57	1.63	0.98	1.38	1.47	0.71	1.17	1.38	0.67
ti180	1.36	0.58	1.65	1.01	1.42	1.54	0.72	1.22	1.39	0.68
ti120	1.36	0.57	1.64	0.99	1.4	1.5	0.72	1.19	1.4	0.69
ti360	1.37	0.59	1.67	1.03	1.45	1.58	0.74	1.25	1.41	0.7
ti720	1.38	0.6	1.69	1.05	1.47	1.64	0.75	1.29	1.42	0.71
ti1440	1.41	0.62	1.74	1.11	1.54	1.83	0.78	1.45	1.46	0.75

Table 5.12: User measures for ti sessions regarding visited content. Abbreviations: AM = Arithmetic Mean; SD = Standard Deviation.

but for the longer timeouts it drops to 20.05% and 18.2% for the 720- and 1,440-minute timeout sessions respectively. This is a clear indicator that users may visit the system twice a day; once in the morning and once in the evening and, regarding the latter, on consecutive days. This is also somewhat reflected in the lead-ins with two per session on average for the higher timeouts, but between two and 1.5 for the lower timeouts.

With this knowledge in mind, one might expect to see greater differences in relation to the content visited, but the differences are comparably small and similar to the **tf** sessions. Table 5.12 displays these measures. The differences between the session variants are straightforward and similar to those seen for the **tf** sessions. The variance for the visited categories on average is small, the standard deviation underlines that. Here, the most interesting observation is that, seeing as even the five-minute timeout has an average of 1.49 visited categories per session, there is a non-trivial number of users that visit different categories in the same session. The slightly lower number of topics per session could be an indicator of logically and contextually related sessions though, especially since the number of visited root categories on average look very similar. The number of products visited resembles the situation just drawn about the categories. The number of queries per session is again relatively low for all session variants (and similarly low level).

Table 5.13 puts this in relation to the time spent in these sessions and the inter-interaction time – there are no surprises. The **ti1440** sessions are the only real outlier because they seem to span multiple days, thereby connecting interactions that would be two sessions for all other variants. Because of this, the average time in session and naturally the inter-interaction time is comparably way higher. The same is true for the **ti720** sessions, spanning morning and evening again, but to a lesser extent. Again, an interesting observation is the comparably low inter-interaction time for all approaches below **ti720** (and even including **ti720** to an extent). All these approaches have a rather low inter-interaction time of less than 10 minutes on average (and also seemingly low standard deviations).

Generally, the inter-interaction time and the time in session is rather low. This indicates that the majority of users stay only a certain amount of time on the system – measuring the distribution here could hint at the look of a global maximum session length designed to fit the underlying data. Knowing what percentage of users stay in the same time buckets makes it possible to estimate how well the inactivity timeout captures user behaviour and to gauge the level of uniformity of this behaviour globally across the system. For now, it suffices to say that there are notable dissimilarities between the different variants.

	∅Time in session		∅Inter-Interactiontime		∅Interaction days	
	AM	SD	AM	SD	AM	SD
ti5	1.43	2.26	0.69	0.56	1.0	0.02
ti10	2.34	3.66	0.97	1.07	1.0	0.02
ti15	3.04	4.82	1.19	1.5	1.0	0.02
ti26	4.21	6.96	1.56	2.35	1.0	0.03
ti30	4.57	7.65	1.68	2.64	1.0	0.03
ti45	5.72	10.02	2.08	3.66	1.0	0.03
ti60	6.69	12.2	2.43	4.64	1.0	0.03
ti90	8.46	16.57	3.08	6.62	1.0	0.03
ti120	10.1	20.96	3.7	8.59	1.0	0.04
ti180	13.29	29.7	4.91	12.6	1.0	0.04
ti360	21.86	52.23	8.31	24.17	1.0	0.05
ti720	39.63	107.42	15.31	48.97	1.01	0.1
ti1440	123.47	361.19	41.01	124.38	1.08	0.28

Table 5.13: User measures for ti sessions regarding time spent. Abbreviations: AM = Arithmetic Mean; SD = Standard Deviation.

Again, an exception is seen in the interaction days; even knowing that an inactivity timeout allows unlimited session lengths, it is still interesting to see that most of the sessions are happening in a single day. Here, the inactivity timeout seems to work and delivers a global result for all sessions.

Unsurprisingly, the top five sequences in Table A3 are no different from the **tf** sessions or visits either. The total share (around 30%) for the top five is the same, as are the type of sequences when comparing it to the **tf** sessions or the visits. There is not that much difference here, but still some interesting observations can be made.

Again, the usual sequence is the single lead-in click on the product page. Taking into consideration that this is still 14.6% for the 1,440-minute inactivity timeout, it is safe to say that this is the most common sequence for users. It is actually quite interesting that, despite queries only being present in about 15% of all interactions, a single query to the system appears to be a common sequence nonetheless. Of the identified sessions for **tf1440**, 5.76% are a single lead-in query, which is the base for all other inactivity sessions as well (with a rising share of sessions for the variants with a lower threshold). Interesting and contrary to the session-identification approaches discussed above, the third position is already a two-click sequence for all variants. With between 3.5% and 4.75%, these sessions seem to make up a good portion of actual user behaviour. Considering that this is a lead-in click on a product page followed by a *leadout*, it is safe to assume these two interactions are related to the same information need. It will be interesting to see if these show up in any of the logical sessions, and how large the share is there.

This concludes the section for the fixed inactivity timeouts. Overall, there were no real surprises here. The biggest differences among the variants are actually seen in the number of sessions and, therefore, in the number of sessions and interactions per user. The time measures mostly correlate with this and the value of the inactivity threshold. Interestingly, the content measures do not change that much, while the inter-interaction time is more or less insignificantly small across the different sessions. This indicates several points:

- There could be a global inactivity timeout that captures the majority of user behaviour with regard to working on one information need

- Users actually work mostly on a limited set of topics during a session
- Temporal inactivity and maximum session length are closely related

It will be interesting to see how the numbers change when looking at the logical approaches. The next step is to see if they change with a dynamically calculated timeout, which is the discussion in the next section.

Dynamic inactivity timeout

Sessions detected with a dynamic timeout use the same assumptions about user behaviour as already described: that a user works on a specific topic for a certain amount of time, which is identified by an inactivity gap that basically reflects the time that passes before work on a new topic begins. Contrary to the fixed inactivity sessions which assumes a global timeout, the dynamically calculated threshold is supposed to ensure that certain characteristics of the system or the user base – depending on the algorithm – is taken into account.

The approaches tested in this dissertation are all based on system specifics, with every threshold calculated on a certain set of conditions. These conditions are combinations of system parameters: visited page types, categories or seasonalities in the form of an interaction day. Every session algorithm has a variance of thresholds depending on the actual content the user visits at the respective time. For example, the **tdpc** approach calculates thresholds based on visited **page_template** and **category_id**, so these would be likely to have different timeouts for interactions on a product page in the category *E-Gitarren* (electric guitars) compared to the category *Katzenfutter* (cat food).

	#Sessions	CV-R	B-R
tdc	319,449,442	78.23%	33.86%
tdcm	319,460,083	78.23%	33.86%
tdp	320,740,004	77.92%	33.93%
tdpc	320,865,062	77.89%	33.87%
tdpcd	322,025,620	77.61%	33.71%
tdpd	321,777,937	77.67%	33.77%
tdpm	320,732,793	77.92%	33.92%
tdpr	320,760,829	77.91%	33.9%
tdr	319,385,497	78.25%	33.9%

Table 5.14: System measures for td sessions. Abbreviations: CV-R = Conversion Rate; B-R = Bounce Rate.

Table 5.14 shows the system numbers for the different **td** sessions. The numbers here come as quite a surprise: taking the variance between the different session-identification approaches, the numbers at this scale are so low that the differences are almost negligible – between the number of sessions as well as the conversion and bounce rates. This is a highly interesting observation, since it essentially means that the calculated dynamic thresholds are either all very similar or are all in the exact same range that they capture user behaviour uniformly, meaning that either the features in use are meaningless or not decisive enough. This is a strong argument for a global inactivity timeout, assuming the different variants actually identify the same sessions and do not just result in the same overall number of sessions by chance.

The bounce rate gives an indication of this because it counts the number of sessions with only one interaction. Still, the ratio is more or less the same with only minor differences between the approaches. This indicates that the majority of single-event sessions are the same here, although this will become even clearer when looking at the numbers on a **user_id** basis.

Curiously, all the measures here are almost identical as well. Table 5.15 displays the system-related measures on a **user_id** and **session_id** basis. The average number of sessions is between 4.08 and 4.11 per user with an average 4.53 to 4.61 interactions per identified session. Logically, because apparently all approaches identify the same sessions, the conversion rate, bounce rate and the number of lead-ins per session do not change in unison.

	⌀Sessions		⌀Interactions		CV-R		B-R		Lead-ins	
	AM	SD	AM	SD	AM	SD	AM	SD	AM	SD
tdc	4.08	11.38	4.6	4.58	90.07%	158.23%	24.73%	31.74%	1.77	1.29
tdcm	4.08	11.38	4.6	4.58	90.05%	158.17%	24.73%	31.74%	1.77	1.29
tdp	4.09	11.58	4.58	4.53	90.02%	158.27%	24.8%	31.78%	1.76	1.29
tdpc	4.09	11.57	4.57	4.5	89.79%	157.62%	24.82%	31.77%	1.76	1.29
tdpcd	4.11	11.73	4.53	4.4	88.98%	155.61%	24.87%	31.74%	1.75	1.28
tdpd	4.11	11.74	4.55	4.44	89.24%	156.33%	24.83%	31.74%	1.76	1.28
tdpm	4.09	11.57	4.58	4.53	90.01%	158.22%	24.8%	31.78%	1.76	1.29
tdpr	4.09	11.58	4.58	4.53	89.94%	158.07%	24.8%	31.77%	1.76	1.29
tdr	4.08	11.4	4.61	4.61	90.23%	158.75%	24.71%	31.74%	1.77	1.3

Table 5.15: User measures for td sessions regarding system usage. Abbreviations: CV-R = Conversion Rate; B-R = Bounce Rate; AM = Arithmetic Mean; SD = Standard Deviation.

The number of average sessions and interactions is located somewhere in-between the **ti30** and **ti45** sessions, indicating a potential global inactivity timeout somewhere between 30 and 45 minutes. This is also true for other measures; the bounce rate as well as the number of lead-ins per session is very close to the aforementioned fixed inactivity approaches. The same checks as outlined above were made again to verify the extent of overlap between these sessions.

Just as the analysis discussed above between the **ti** and **tf** sessions, the overlap between the session approaches is very high. The **ti30** sessions in particular are almost identical to all the **td** approaches with only minor differences, with 97.63% of all interactions sharing the same **session_id** between **ti30** and the **td** sessions. With knowledge of this overlap, it makes sense to review the timeout statistics to see what can be revealed about the actual timeouts from the data.

	Maximum	Minimum	AM	SD	Median
tdpd	636.6	0.17	38.92	56.73	29.15
tdpm	291.39	0.31	35.38	30.93	29.26
tdr	78.14	0.18	38.92	15.05	38.21
tdcm	1, 290.94	0.0	37.2	18.68	37.8
tdpcd	1, 435.65	0.0	37.12	64.84	27.89
tdpr	189.47	0.02	29.38	24.0	26.58
tdc	184.38	0.14	36.97	12.23	37.63
tdpc	1, 432.4	0.0	33.8	46.88	29.84
tdp	142.79	0.31	36.16	29.14	29.68

Table 5.16: Descriptive overview of td timeouts. Abbreviations: AM = Arithmetic Mean; SD = Standard Deviation.

Table 5.16 shows the descriptive statistics relating to the calculated time values per dynamic inactivity approach. The average is between 30 and 40 minutes with a rather fluctuating standard deviation. The mean is also around 30 minutes. The minimum and maximum values are not really meaningful; the minimum values are all below one minute while the maximum values are also very widespread. An interesting observation is that the maximum for the **tdr** sessions is rather low compared to the other session-identification approaches with a maximum of about 78 minutes. Considering that the average is similar, it is safe to assume that the majority of the values are also similar.

In any case, these inactivity gaps do not have to mean much. The interesting part is how they get applied to the interaction data, which is most likely the cause for the very similar identification results. If the majority of interactions is on the product page (**page_template** *OffersOfProduct*) in the category *Smartphones* (**category_id** *19116*), all of these interactions are connected to sessions with the same or at least a very similar timeout. The assumption is that neither the used device or seasonality in the form of the month of the interaction have any real impact on the average time spent in the respective combination of dimensions, leading to a very similar timeout across all approaches and, more importantly, across all interactions in the data itself. Some features are not meaningful, others are applied predominantly in the dataset.

The remaining measures for the **td** sessions can be found in Table A4 and Table A5. Considering that the measures regarding the visited content are also nearly identical across all session approaches, the distribution of **page_templates** and categories is most likely the cause for the observed results. All the dynamic-inactivity approaches have an average of 1.34 root categories, 1.59–1.6 categories, 1.32–1.34 products, 0.7 queries and 1.36 potential topics per session. The visited content is more or less the same with a few minor differences. The number of 1.36 potential topics (and the values for the categories, naturally) could be an indicator that the session boundaries are being too liberally detected. Then again, this is also very similar to the **ti** sessions, which show the same quantity. Logically, the numbers related to the session length and inter-interaction time are also nearly identical. Again, this is close to the **ti** sessions with an average time in session of around about 5.3 minutes and an average inter-interaction time of about two minutes.

It is quite a surprise that the sessions with a dynamic timeout are, in the end, very similar to the fixed timeout sessions with an apparent global timeout of somewhere between 30 and 45 minutes. As the test showed, the overlap between these session approaches is relatively high, indicating that the majority of the **td** sessions are indeed identified with roughly the same timeout which has to be around 35 minutes. This means, a 35-minute timeout seems to be applicable here no matter the actually visited content (disregarding the small differences at this point). This is an interesting observation, because it points to a 35-minute global value that is more or less identical for the whole system and population. With this knowledge in mind, it is now time to take a look at the logical session approaches and see how the session identification worked out here.

5.2.2 Logical Sessions

This section is dedicated to the logical sessions. Having given an overview of the mechanical sessions, now the various logical approaches without any temporal limit or mechanical boundary will be analysed. With knowledge of the significant differences between the mechanical sessions, it will be interesting to see how the logical approaches will identify sessions and to compare and contrast the differences here.

The first section takes a look at the lexical sessions. As their definition is aligned closely with mechanical sessions in this dissertation, the lexical sessions will be analysed first to see how big the differences are. Afterwards, the sessions identified by using the retrieved similar categories using the BM25 algorithm are analysed. The final section will describe logical sessions using category vectors.

5.2.2.1 Lexical Baselines

The first variant of the logical approaches is actually a mixture of logical assumptions with a somewhat mechanical execution, at least in this dissertation. Just to recap, lexical similarity is defined here as a match between categories of interactions; two interactions i_n and $i_n + 1$ belong to the same session, when the **root_category_id** of these interactions is identical. Otherwise, a new session begins.

With this session approach, an additional dimension is introduced that was not present in the mechanical sessions. The two lexical variants are identical in their comparison method, but the comparison context is now different: the first one (**lcdb1**) looks at consecutive events to compare them, just like all the mechanical approaches did up to now; the second approach (**ladb1**) takes interleaving behaviour into account and compares not only consecutive events but also all previous events with a reference event. While lcdb1 compares i_n with $i_n + 1$, ladb1 uses the last event of all previous sessions as a comparison base for i_n, allowing for interleaving patterns. As neither of the approaches takes time passing into account, theoretically, therefore, both enable sessions of unlimited length.

	#Sessions	CV-R	B-R
lcdb1	229,523,765	108.89%	28.96%
ladb1	145,886,471	171.31%	20.74%

Table 5.17: System measures for lexical sessions. Abbreviations: CV-R = Conversion Rate; B-R = Bounce Rate.

An important time-related element to consider here are interactions that happen at exactly the same time. The timestamp in the data used for chronological ordering (and highly relevant in all logical approaches, therefore) are millisecond precision. When two interactions have the same timestamp (which can happen due to the granularity of milliseconds), this leads potentially to differences when trying to reproduce the sessions since the ordering of events is non-deterministic. This affects around 37,000 of the 78m **user_ids**. The implications of this will be discussed at more length in Section 5.3; for now, it is enough to know that reproducing the same numbers with the current configuration of the logical algorithms is a random matter. It is likely that there will be a certain margin of difference, although probably a small one in actuality, in the resulting sessions[1].

Table 5.17 shows the system measures for both tested approaches. The difference in the number of sessions is immediately apparent since it is very high. While the direct comparison between consecutive sessions identifies 229m sessions, the approach taking interleaving

[1] During the test runs in this dissertation, the resulting sessions for the baseline approaches differed by less than 1,000 sessions in different calculation runs.

behaviour into account detects considerably less sessions with only 145m sessions. From a numbers' perspective, this makes the direct comparison between consecutive events comparable to the **ti1440**. The conversion rate and bounce rate are also relatively, with a difference of only two percentage points.

Looking at the overlap between these two approaches, the differences are notable though. There seem to be around 100m sessions that are differently identified; the two approaches do not identify the same sessions (although there is of course an overlap of interactions in the same sessions). This is an interesting finding. Apparently, both approaches identify a comparable number of sessions but a huge batch of these are differently structured. From this, it could be surmised that there are interaction sequences that occur over the course of one day focused on one category branch, while on other days there are sequences that may touch different branches. It would be interesting to know whether this is a general behavioural pattern or if it's only observable for certain users.

	≤ 10		$>10, \leq 30$		$>30, \leq 100$		$>100, \leq 500$		>500	
lcdb1	83,795,007	36.51%	50,651,937	22.07%	43,169,997	18.81%	36,705,793	15.99%	15,201,031	6.62%
ladb1	78,139,912	53.56%	37,198,503	25.5%	20,426,076	14.0%	8,690,224	5.96%	1,431,756	0.98%

Table 5.18: Share of lexical sessions per interaction bucket. Displayed are the absolute number of sessions and the share in percent.

The session approach using the non-consecutive comparison context creates fewer sessions; the numbers here are not comparable to any of the previous approaches. The conversion rate is logically higher, because the number of sessions is that much lower. The bounce rate is very interesting though, since it is still on a similar level when comparing it to some of the mechanical approaches; around 20% of all identified sessions are bounces. Considering that there are still 145m sessions overall, about 30m apparently consist of interactions disconnected from any other interaction (of the respective **user_id**).

Table 5.18 breaks down the number of identified sessions per interaction bucket to look at the size of the variances. The distributions across the buckets varies a lot between the two session approaches. The table shows the absolute sessions as well as the share of all sessions per approach and per bucket. The most interesting observation is the behaviour of the higher interaction buckets. The bucket with the highest number of interactions (more than 500) has only slightly over 1m sessions overall, which is equal to just 1% of the total sessions for the second approach. Bearing in mind that these are sessions of around 150,000 **user_ids** with around 137m interactions, this seems quite a low number. The same is true for the bucket with more than 100 and less or equal to 500 interactions, but here it could be assumed that users tend to have very long sessions identified by the second approach, where they visit the same category branch repeatedly over time, working on the same information need. The first bucket with less than or equal to 10 interactions looks quite normal in comparison; the total number differs only slightly from the direct comparison between consecutive events, which is logical considering that there is a limit to the number of comparable events. If they all belong to the same category branch, both approaches will identify very similar or even identical sessions. In the second approach, the categories seem to get more diverse as the number of interactions rise, resulting in the identification of fewer sessions.

When the first approach is compared to its similar mechanical counterpart (**ti1440**), again there is a surprisingly high similarity. The first interaction bucket is nearly identical when comparing the number of sessions. With the exception of the last bucket, where the lexical approach seems to identify slightly more sessions, the other buckets also show very similar numbers. This is once again very logical, bearing in mind the number of interactions used for comparison here. It is likely that the time constraint will connect more sessions than the lexical approach in such a scenario.

	∅Sessions		∅Interactions		CV-R		B-R		Lead-ins	
	AM	SD	AM	SD	AM	SD	AM	SD	AM	SD
lcdb1	2.93	8.08	5.94	7.53	118.08	215.98	17.56	28.12	2.42	2.82
ladb1	1.86	1.46	7.25	11.03	143.91	270.11	14.2	26.11	2.92	3.86

Table 5.19: User measures for lexical sessions regarding system usage. Abbreviations: CV-R = Conversion Rate; B-R = Bounce Rate; AM = Arithmetic Mean; SD = Standard Deviation.

Table 5.19 compares both approaches on a **user_id/session_id** level. Again, the differences and the apparent similarity of the first approach to the **ti1440** sessions are very interesting. The average values of both approaches are very close with the lexical approach identifying slightly fewer sessions. Of course, this has an impact on the other values, because the average interactions per session are slightly higher. The conversion rate, bounce rate and the number of lead-ins follow the same trend. This is interesting considering the previously noted difference in the identified sessions – so it would seem that the actual structure of the sessions remains similar in terms of behaviour.

The second approach is very different in comparison. While the average number of sessions is below two per user, which is surprisingly low, the average number of interactions is logically higher: with 7.25 interactions on average and a low bounce rate of only 14.2%, this approach identifies rather long sessions with a low number of bounces; then again, considering that the bounce rate for the first approach is 17.56%, the 14.2% bounces do not seem so low. Essentially, to explain the nature of the bounced sessions, these focus on one topic and are never picked up again, which is why the sessions are not continued. Breaking these down among the interaction buckets, there is little difference across the different buckets; they all have around 14% bounces, which is surprising but, in a way, logical; most likely, the users are visiting a more or less disconnected root category or category during these sessions. Looking at the data again, the majority (about 18.7m) of the bounced sessions come from **user_ids** with less than 10 overall interactions. Considering that there is a limit to the numbers of categories that can be visited during 10 interactions, this seems reasonable. The remaining bounces in the higher interaction buckets are more interesting; a likely scenario here would be that users make use of the system for a limited number of root categories rather than for every topic or information need they have, resulting in single interactions on topics that are not regularly worked on.

	∅Root Categories		∅Categories		∅Products		∅Queries		∅Topics	
	AM	SD	AM	SD	AM	SD	AM	SD	AM	SD
lcdb1	1.23	0.39	1.66	0.91	1.64	1.9	0.72	1.09	1.21	0.46
ladb1	1.24	0.4	1.85	1.14	1.89	2.34	0.86	1.58	1.32	0.6

Table 5.20: User measures for lexical sessions regarding visited content. Abbreviations: AM = Arithmetic Mean; SD = Standard Deviation.

Table 5.20 shows the differences between the two lexical approaches regarding the visited content. Theoretically, the measures here should differ only minimally considering the nature of the comparison method – both utilize matching **root_category_ids** to connect interactions to sessions, thus only differing in the comparison context. This assumption holds at least somewhat true, with the second approach showing only slightly higher average values. Again, this makes sense because of the algorithm's potential to connect a greater variety of categories to the same session, whereas in the first approach using direct comparison, potential interleaving behaviour may cause the session to end prematurely.

The number of potential topics between the two approaches is not so different, with slightly higher potential topics for **ladb1**. This is reasonable for the same rationale as outlined above; as there are more opportunities for **ladb1** to include more categories throughout a session, the possibility of more topics is greater. The average number of potential topics being 1.21 / 1.32 is curious in itself though, particularly in relation to the **lcdb1** sessions; here, one would assume that consecutive interactions with the same root category belong to the same topic, which implies that the **lcdb1** sessions have exactly one topic. As this is definitely not the case, it must be presumed that the related categories identified as potential topics do not necessarily belong to the same root category – which is expected behaviour but also somewhat surprising as it means that categories in the same root branch are not necessarily topically connected. A further observation is the similarity in numbers between the mechanical sessions and the topics as well as root categories – that these do not differ greatly is an indicator that the mechanical approaches are quite close to being able to connect related interactions.

	∅Time in session		∅Inter-Interactiontime		∅Interaction days	
	AM	SD	AM	SD	AM	SD
lcdb1	11, 293.31	40, 268.45	3, 775.46	17, 859.33	1.42	1.14
ladb1	23, 464.53	57, 474.03	6, 383.9	22, 357.26	1.69	1.72

Table 5.21: User measures for lexical sessions regarding time spent. Abbreviations: AM = Arithmetic Mean; SD = Standard Deviation.

Both approaches have no time constraint in theory. They could very much go on indefinitely. Table 5.21 reflects this notion, whereby the average values for time in session and inter-interaction time are very high, indicating sessions spanning multiple days or even weeks. In relation to the first approach, this points to yet another very clear divergence in the **ti1440** sessions. That the time in session and the time between interactions is much higher, indicates once again that they do not necessarily identify the same sessions, rather only a similar number of sessions with similar content characteristics.

The second approach is even freer with these measures; here, there is no time constraint and no constraint regarding the comparison context between events, so sessions may span the whole year. The average of 23,000 minutes (around 16 days) with a standard deviation of 57,000 (around 40 days) may indicate just that. Naturally, the inter-interaction time is also very high with around 3,700 minutes for the first approach and 6,300 minutes for the second approach. The interaction days are another clear indicator, although the average here is lower. It can be assumed that while the time in session and the inter-interaction time is somewhat disturbed by outliers, it is still reasonable considering the nature of the algorithms.

To underline this idea, as the second approach is not constrained by consecutive interactions it is interesting to note how other sessions may interrupt time spent in a session or the inter-interaction time. To measure the impact of this interleaving behaviour, the session breaks for this approach were counted. Overall, 17.98% of all **ladb1** sessions show interleaving behaviour, with an average of 2.84 breaks of interaction sequences (1.94 on **user_id** basis). This means that, on average, these sessions usually consist of around four separate sequences. Still, the number of sessions that show no interleaving behaviour at all is relatively high. Considering that interactions are connected via **root_category_id** at any place on the timeline of a user's history, the share seems rather low. This could indicate multiple things: users tend to focus on a small set of topics in their interactions with the system, especially in the lower interaction buckets, or shopping journeys may take way longer than expected; it may very well be that the time frame of a year is not really long enough to grasp a full view of the user's interests – these users may work exclusively on one topic before finally changing to another.

The sequences in Table A3 show the differences between the two approaches very well. Overall, the sequences are similar to the general sequences, but the shares are noticeably smaller compared to the other mechanical approaches. The bounced sessions still make up the biggest share. Another interesting observation is that the **ladb1** approach creates 1.7% sessions with a lead-in on the homepage only; this is a clear weakness of the algorithm since it cannot match pages without any category (like the homepage) and therefore does not connect them if they are more than 24 hours from any other trace with a category. Therefore, these are single clicks on a specific day with nothing else happening there.

Another potential breaking point are the query pages; these usually do not come with a **category_id**, the **category_id** in the data is only added during preprocessing using an IR algorithm and inventory data. This matching between **category_ids** and queries may lead to errors in a sequence; for example, a session break may occur when a user works on a topic using queries and one of the queries is assigned an incorrect **category_id**. This would lead to session breaks. The relatively high number of session breaks in the lexical approaches could indicate the possibility of such errors.

This concludes the section on the lexical approach. A number of interesting observations have emerged regarding the potential boundaries of lexical connections and how strict matching between **category_ids** can potentially lead to longer sessions. Nonetheless, it seems likely that users do use the system as suspected: working on a specific topic for some interactions before switching to the next topic after a break. The differences in numbers between the two approaches are interesting as well since the two comparison contexts apparently lead to very different sessions. Curiously enough, the visited-content measures stay similar. It will be quite intriguing to see how these numbers might change when the comparison between categories is even more liberal. This is the task of the next section, which will show what happened when the sessions were identified by comparing the matched-term vocabulary using the BM25 retrieval algorithm.

5.2.2.2 Shared-Term Space using BM25

This section reveals how the four different session approaches were performed using similarity calculated by the shared-term space between categories, calculated and retrieved via the BM25 retrieval algorithm. Note that the comparison method is the same for all approaches and that only the comparison contexts differ. To recap on this, there are four different comparison contexts: direct comparison between two consecutive interactions; comparison between a reference interaction and all interactions of the session prior to the reference interaction; direct comparison between a reference event and the last event of all previous sessions; and lastly, the comparison between a reference interaction and all interactions of all previous sessions[2].

Here, the basic assumption is the same as for all logical sessions: users work on a single topic, using interleaving behaviour, potentially over a longer period of time. However, there are slight differences in assumptions between the different comparison contexts. The consecutive comparison assumes that users work in small portions of interactions on the same topic. When directly comparing two consecutive interactions, the assumption is that the topic will probably evolve during a session, thus, only the last interaction is important when connecting new interactions to the current session. Conversely, for the comparison between all events, it is assumed that all sessions that relate to the same topic belong together.

	#Sessions	CV-R	B-R
bm25cd	308,865,935	80.91%	34.55%
bm25cc	299,628,724	83.41%	34.89%
bm25ad	231,283,812	108.06%	27.76%
bm25ac	215,728,569	115.85%	28.72%

Table 5.22: System measures for bm25 sessions. Abbreviations: CV-R = Conversion Rate; B-R = Bounce Rate.

The consecutive sessions differ from the interleaving sessions. The consecutive comparison identifies bursts of interactions belonging to the same topic. Sessions belonging to the same topic may occur multiple times, when the user decides to work on a certain topic again. Connecting all these related sessions may lead to journeys. The interleaving sessions enable immediate identification of journeys: instead of detecting and separating granular sessions focusing on a specific

[2] For simplicity, the four comparison contexts are put into two groups: consecutive and interleaving. Both terms are used synonymously for the respective contexts.

task, all interactions contextually related in any way are connected. This is important to keep in mind when comparing the numbers.

The assumption for the comparison method using the shared-term space is based on the simple idea that topically related categories will share a certain vocabulary. For example, the category *Smartphones* will share many common words with contextually related categories such as *Handytaschen* (mobile phone cases), *Displayschutzfolien* (screen protectors) or even *Kopfhörer* (headphones). This may lead to some potential errors depending on the category, but the general idea is that only categories that are at least somewhat related should be connected. However, there is potential for an overly liberal connection of categories.

Table 5.22 displays the top-level measures for the four variants. The table shows clear differences in terms of session numbers. Both consecutive approaches identify far more sessions than the lexical baseline approaches. They have a comparable number of sessions as the **ti60** to **ti120** sessions but it can be assumed that the actually identified sessions are structurally different. The two approaches allowing for interleaving behaviour produce more sessions, somewhat comparable to the **ti1440** sessions and the consecutive lexical sessions, but less than the other lexical approach. Apparently, the BM25-based term-matching is stricter than the matching of categories. This would somewhat make sense, considering that the lexical sessions take the **root_category_id** into account, while the **bm25** matching is more fine-grained (although still very liberal). Logically, having a stricter set of rules results in more sessions.

The conversion rate follows the session trend. Again, the bounce rate is more interesting, this stays more or less similar. Despite the differences between the approaches, the bounce rate is somewhere around 30%. Almost one-third of the identified sessions consist of only one interaction, no matter the comparison context. This is likely to be because these interactions are categories with few other similar categories – this would be the case for more specific categories. Another possibility again may involve interactions on the homepage that cannot be associated with any category from a similarity point of view and are more than 1,440 minutes away from any other interaction, having an impact on the bounce rate. It seemed unreasonable initially bearing in mind the broader comparison context[3] that the bounce rate for the **bm25ac** sessions is higher than for the **bm25ad** sessions, but the absolute numbers for both approaches are reversed.

	∅Sessions		∅Interactions		CV-R		B-R		Lead-ins	
	AM	SD	AM	SD	AM	SD	AM	SD	AM	SD
bm25cd	3.94	13.25	4.66	5.07	95.18%	173.13%	24.22%	31.2%	1.94	2.02
bm25cc	3.82	12.53	4.78	5.34	97.51%	177.72%	24.05%	31.25%	1.98	2.1
bm25ad	2.95	5.29	5.09	5.45	103.55%	180.08%	21.4%	29.66%	2.1	2.18
bm25ac	2.75	3.69	5.32	6.02	107.69%	188.19%	21.28%	29.72%	2.17	2.34

Table 5.23: User measures for bm25 sessions regarding system usage. Abbreviations: CV-R = Conversion Rate; B-R = Bounce Rate; AM = Arithmetic Mean; SD = Standard Deviation.

[3] Using all interactions from all previous sessions for comparison should result in more connected sessions compared to using only the last interaction of all previous sessions.

Table 5.23 breaks the system usage down on a **user_id** level. The number of sessions per user decreases as the number of sessions decreases. Likewise, the interactions per session increase. The same is true for the conversion rate and the bounce rate; the former increases and the latter decreases. The number of lead-ins increases for the non-consecutive comparison contexts. As visible, the conversion rate may rise above 100% due to the way it is calculated (as described in Section 4.5 using the total number of lead-outs instead of a binary encoding of the transaction); this also explains the high standard deviations, apparently, some sessions have a high number of lead-outs in comparison to others.

Comparing the numbers to the lexical variants again, it is obvious that the **bm25** sessions use stricter rules. The similar categories according to the **bm25** ranking are apparently different to the connection via root category. Unsurprisingly, the overlap between the two- and four-session approaches is rather low; the identified sessions are structurally different although the numbers regarding system usage are somewhat comparable. The same is true for the mechanical sessions; the overlap seems even smaller. The identified sessions, therefore, very much differ from the previously analysed approaches.

	∅Root Categories		∅Categories		∅Products		∅Queries		∅Topics	
	AM	SD	AM	SD	AM	SD	AM	SD	AM	SD
bm25cd	1.21	0.37	1.34	0.5	1.33	1.42	0.56	0.75	1.02	0.12
bm25cc	1.22	0.38	1.35	0.53	1.35	1.47	0.57	0.79	1.03	0.14
bm25ad	1.23	0.38	1.38	0.53	1.4	1.47	0.6	0.79	1.04	0.14
bm25ac	1.23	0.39	1.4	0.56	1.44	1.55	0.62	0.86	1.04	0.15

Table 5.24: User measures for bm25 sessions regarding visited content. Abbreviations: AM = Arithmetic Mean; SD = Standard Deviation.

Table 5.24 shows the measures in relation to visited content. These are again quite surprising when the aforementioned system numbers are considered. There are almost no differences at all between the session approaches, which is another quite surprising finding, considering the interleaving approaches should have the potential to encounter many more categories. This is truer still for the actual comparison base; bearing in mind that these session approaches use the complete session (i.e. all categories and their similar categories) when comparing a reference event, they should see more distinct categories per session on average.

One explanation for this could be that the similarities between categories extracted by the BM25 retrieval algorithm are simply very self-contained: every category has a set of similar categories that is the same throughout all of these similar categories. For example, the category *Smartphones* would have n similar categories with the category *Handytaschen* (mobile phone cases) among them. Both categories would be likely to share a high overlap between their n similar categories, indicating that the **bm25** session identification would lead to very strict logical sessions focusing on very specific tasks. This is the clear distinction that probably belongs with other logical variants also, which follow in the next section.

Again, the time measures in Table 5.25 are comparably quite intuitive, although here one might expect even higher numbers in relation to time spent in sessions for the approaches comparing all previous sessions with a reference event. The time in session for these is twice as high compared to the consecutive comparison approaches. Comparing this finding to the lexical approaches, the time measures are much lower, indicating that the identified sessions are shorter in general. The inter-interaction time reflects this as well; lower than both the lexical approaches (because of the comparison method) but generally higher than the mechanical approaches.

	∅Time in session		∅Inter-Interactiontime		∅Interaction days	
	AM	SD	AM	SD	AM	SD
bm25cd	3, 918.33	20, 705.99	1, 563.44	10, 537.43	1.23	0.74
bm25cc	4, 099.07	21, 404.76	1, 570.73	10, 532.9	1.24	0.78
bm25ad	7, 456.73	25, 553.76	2, 871.9	13, 767.29	1.31	0.85
bm25ac	8, 087.8	27, 025.94	2, 903.89	13, 796.77	1.33	0.94

Table 5.25: User measures for bm25 sessions regarding time spent. Abbreviations: AM = Arithmetic Mean; SD = Standard Deviation.

However, there are only very minor differences within the respective comparison-context groups. Enabling the identification algorithm to make the comparison among all previous interactions in a session does not seem to make too much of a difference using the BM25 similarities. These observations are further arguments for the self-containedness of these sessions based on their category similarity; the shared-term space extracted by the BM25 ranking seems to create somewhat enclosed-term silos among the top similar categories for a given category.

Again, the top five sequences are more or less similar to the other approaches as well, caused by the fact that the bounce rate is very similar to the other approaches. Table A3 shows no real differences except that the interleaving session approaches already have two-interaction sequences in the top five. Again, it is interesting to see that both these approaches have very similar sequences and shares in the top five, yet again indicating that the bm25-based comparison method is very self-contained regarding category similarity.

Looking at the interleaving behaviour, the values are similar to the lexical sessions. For **bm25ac**, 13.58% of all identified sessions show interleaving behaviour, with an average of 2.3 breaks in the identified sequences. For **bm25ad**, slightly different values can be reported: 14.07% of the identified sessions show breaks with an average of 1.96 breaks. Since the absolute values are much higher compared to the lexical variant from the previous section, the similar percentages may be misleading. Nevertheless, the share of interleaving sessions is slightly higher here. Around 30m of the identified sessions show interleaving behaviour compared to the roughly 26m of the lexical variant. The difference is interesting, but explainable by the actual session breaks – the lexical variant identifies fewer sessions but with more session breaks.

5.2.2.3 Category Embeddings

The final logical approach tested in this dissertation utilizes word2vec to estimate category similarities. Using the same four comparison contexts as before, the difference is the method of comparison. Whereas before simply **root_category_ids** and the similarity as extracted by the BM25 retrieval algorithm were used, the following section deals with approaches using word2vec on the complete user history of interactions to create category vectors. The assumption behind these similarities is based on the same assumption for the original word2vec algorithm presented by Levy and Goldberg [135]: words in close proximity to each other are likely to be related. Therefore, the sequences of consecutive categories are used to create category vectors.

These vectors are then used to calculate cosine similarity between the different categories of the category tree. The resulting values are used to match categories and decide if they are similar, eventually deciding about session continuation or the start of a new session. The distribution of the cosine similarity between categories is hard to interpret because it is based on many different elements (i.e. context window, data structure, further algorithm parameters). To decide if a category was similar to another, three different thresholds were used: top 10 similar categories; all categories above a 0.5 threshold according to the calculated cosine similarity; and a calculated threshold loosely based on the distribution of the data (using the standard deviation). These three methods were tested in all comparison contexts.

A specialty of the logical approaches tested in this dissertation is that the approaches respect changing and evolving information needs. By the phrase 'changing or evolving information needs', a change in the direction of working on a topic is meant; the algorithm will compare categories based on their calculated similarities and assign a **session_id** accordingly. If a new **session_id** is assigned, the respective **category_id** of the reference interaction and all its calculated similar categories will be assigned the same **session_id**. Ultimately, this means that categories can be in multiple sessions although they are topically related because the most recent category is seen as the driving topic and will reassign ids accordingly. For the comparison that follows, the categories will use the newly assigned or reassigned **session_id**. This was not really apparent in the **bm25** sessions due to the nature of the calculated similarities, but will be more prevalent in the vector approaches.

This peculiarity can be seen as a potential weakness of these types of algorithms. It is a design choice that may be handled otherwise; in this dissertation, it is handled this way to put some more weight on the most recent activity and to be able to have evolving shopping journeys. Handling it differently by keeping every **category_id** strictly within one session would be likely to massively reduce the number of identified sessions. Another foreseen downside would be that some categories may continually end up in their own sessions due to their likeness being based on the cosine similarity, and this could lead to different results for different categories even though they are potentially related.

Table 5.26 shows the overall system measures for the vector approaches. The list is ordered by approaches first and then by comparison context respectively. The following tables are ordered likewise. First come the sessions where all categories above 0.5 cosine similarity are taken to be alike; next are the sessions taking the top-10 most similar sessions according to cosine similarity; and lastly, the sessions with the calculated threshold. The comparison contexts are shown as before. That the differences between the comparison contexts are relatively high is very reasonable. The differences in-between these contexts are more or less similar to the **bm25** sessions. Interestingly, the differences between the actual comparison method variants are not that high. Using either the top 10 similar categories, all categories above 0.5 cosine similarity or the ones determined via the calculated cut-off seems to make a noticeable difference when identifying sessions, but the variance between the approaches is rather small across all comparison contexts.

	#Sessions	CV-R	B-R
u2v05cd	302,896,160	82.51%	35.23%
u2v05cc	300,045,032	83.29%	35.27%
u2v05ad	218,918,746	114.16%	28.87%
u2v05ac	213,752,978	116.92%	29.12%
u2v10cd	301,526,828	82.88%	34.54%
u2v10cc	296,737,824	84.22%	34.62%
u2v10ad	221,684,899	112.74%	28.22%
u2v10ac	212,641,685	117.53%	28.77%
u2vccd	293,231,413	85.23%	33.99%
u2vccc	288,285,363	86.69%	34.13%
u2vcad	214,602,352	116.46%	27.77%
u2vcac	205,756,066	121.46%	28.27%

Table 5.26: System measures for u2v sessions. Abbreviations: CV-R = Conversion Rate; B-R = Bounce Rate.

An important observation becomes apparent when ordering by sessions. Usually, the approaches using all categories above the 0.5 threshold of cosine similarity identify the most sessions, followed by the top 10 with slightly less and the cut-off sessions with even fewer identified sessions. In the case of the **ad** sessions – allowing interleaving behaviour but only using the last interaction of a previously identified session for the comparison to a reference object – the order is different; here, the 0.5 cut-off sessions identify slightly less sessions than the top 10 sessions. This observation is hard to interpret, but it hints at structural properties in the dataset that lead to different results. The likely cause of more identified sessions in the top-10 approach is that the last category in certain interaction sequences has more than 10 sessions above the 0.5 threshold. Curiously, when using direct comparison between consecutive interactions, this effect is not apparent, although the difference between the two session approaches is also very small here (only around 1.3m).

It can be surmised that the cut-off sessions identify sessions more liberally than the other approaches, with the 0.5 threshold being the strictest. Analysing the dependencies between the ratio of similar categories for a given category would be a good starting point for improving this type of logical session. As is, the comparison methods are more or less arbitrary but seem to be able to identify a somewhat similar number of overall sessions – fine-tuning this similarity map may lead to a more fine-grained picture, but will most probably result in a comparable number of sessions.

The bounce rate is consistent across the comparison contexts. The share of the bounced sessions is usually the same. Although it seems that the share is higher for the sessions identified using the complete session history when comparing it to the last-interaction comparisons, the absolute numbers are less. This is expected but the share is nevertheless relatively high. The same is true for the approaches allowing interleaving behaviour. It is surprising to find the bounce rate this high considering the nature of these approaches. Looking at the data again and aggregating the identified bounced sessions by **category_id**, it looks like a big part of the bounces come from either unconnectable interactions (i.e. interactions on the homepage more than 1,440 minutes from any other interaction with a meaningful **category_id**) or are sessions by users with a small number of interactions looking at different specific categories.

	∅Sessions		∅Interactions		CV-R		B-R		Lead-ins	
	AM	SD	AM	SD	AM	SD	AM	SD	AM	SD
u2v05cd	3.87	13.04	4.79	5.36	97.6%	178.34%	24.05%	31.28%	1.98	2.09
u2v05cc	3.83	12.75	4.84	5.53	98.41%	180.94%	23.99%	31.29%	1.99	2.15
u2v05ad	2.79	4.23	5.32	5.93	107.63%	187.72%	21.18%	29.67%	2.17	2.3
u2v05ac	2.73	3.68	5.41	6.26	109.2%	192.44%	21.14%	29.68%	2.2	2.4
u2v10cd	3.85	12.91	4.79	5.33	97.52%	177.62%	23.74%	31.12%	1.97	2.08
u2v10cc	3.79	12.48	4.86	5.51	98.85%	180.8%	23.67%	31.14%	2.0	2.12
u2v10ad	2.83	4.54	5.27	5.82	106.77%	185.96%	21.0%	29.58%	2.15	2.27
u2v10ac	2.71	3.59	5.42	6.26	109.41%	192.4%	20.95%	29.61%	2.2	2.38
u2vccd	3.74	12.42	4.9	5.48	99.51%	181.05%	23.07%	30.86%	2.02	2.12
u2vccc	3.68	11.98	4.98	5.74	100.97%	185.03%	22.97%	30.88%	2.04	2.21
u2vcad	2.74	4.28	5.41	6.02	109.22%	190.06%	20.39%	29.32%	2.21	2.33
u2vcac	2.63	3.36	5.56	6.55	112.04%	197.62%	20.34%	29.36%	2.25	2.48

Table 5.27: User measures for u2v sessions regarding system usage. Abbreviations: CV-R = Conversion Rate; B-R = Bounce Rate; AM = Arithmetic Mean; SD = Standard Deviation.

Table 5.27 sets out the user-based system measures for the user2vec session approaches. The various measures for the approaches follow the findings already mentioned. The differences between the approaches within the same comparison contexts are not that high. In terms of the average session per user, the difference between the approaches ranges from 0.10 to 0.15 less sessions on average when compared to the 0.5 threshold sessions with the cut-off variants. The difference is somewhat steady across all the comparison contexts. One interesting observation is found in the standard deviations; the deviations of the consecutive approaches are far greater than those that allow interleaving behaviour. This is reasonable bearing in mind that the interleaving approaches connect interactions easier than the consecutive ones, but it is still an important clue about user behaviour. Apparently, there are users working consistently on the same topics without interruption (allowing the consecutive approaches to connect all the interactions) whereas other users switch relatively often between topics (causing a new session with every interruption). This observation is not apparent in the interactions though, which may indicate that users make the same number of interactions – either on the same categories or with new ones. The interactions per session are also very closely aligned across the comparison contexts as between the interactions too, with 5.56 interactions for the **u2vcac** sessions being the highest and 4.79 being the lowest value for the **u2v10cd** or **u2v05cd** sessions.

The conversion rate and bounce rate do not differ significantly in comparison to the overall system measures. The average bounce rate per user is lower, indicating that the majority of overall bounced sessions come from users that have a low number of overall interactions. With the bounce rate at 24% to roughly 20%, however, it is still relatively high. It will be interesting to see if the temporal component changes that rate for the combined approaches discussed in the following section. The number of lead-ins per session is also not very surprising considering that it is a fixed entity that will change depending on the length of the identified sessions. The longer sessions have logically more lead-ins on average. The standard deviation here is very consistent across sessions, indicating again that there is not a strong difference in session length across the approaches and comparison contexts (despite the normal differences that were discussed).

	⌀Root Categories		⌀Categories		⌀Products		⌀Queries		⌀Topics	
	AM	SD	AM	SD	AM	SD	AM	SD	AM	SD
u2v05cd	1.21	0.37	1.36	0.55	1.35	1.47	0.57	0.78	1.0	0.03
u2v05cc	1.21	0.38	1.37	0.58	1.36	1.5	0.57	0.81	1.0	0.03
u2v05ad	1.23	0.38	1.4	0.57	1.43	1.53	0.62	0.84	1.01	0.04
u2v05ac	1.23	0.38	1.41	0.6	1.45	1.58	0.63	0.88	1.01	0.05
u2v10cd	1.22	0.38	1.36	0.55	1.35	1.46	0.57	0.77	1.01	0.06
u2v10cc	1.22	0.38	1.37	0.57	1.36	1.5	0.57	0.8	1.01	0.06
u2v10ad	1.23	0.38	1.4	0.57	1.43	1.52	0.61	0.82	1.01	0.07
u2v10ac	1.23	0.39	1.42	0.6	1.46	1.58	0.63	0.88	1.01	0.07
u2vccd	1.22	0.38	1.39	0.59	1.38	1.49	0.58	0.81	1.0	0.0
u2vccc	1.23	0.39	1.41	0.62	1.39	1.54	0.59	0.84	1.0	0.0
u2vcad	1.23	0.39	1.44	0.61	1.46	1.55	0.63	0.86	1.0	0.0
u2vcac	1.24	0.4	1.46	0.65	1.49	1.63	0.65	0.93	1.0	0.0

Table 5.28: User measures for u2v sessions regarding visited content. Abbreviations: AM = Arithmetic Mean; SD = Standard Deviation.

Table 5.28 shows the content measures such as the average number of visited categories per session approach. Considering that all three comparison methods are based on the **category_ids**, these numbers provide information about the effectiveness of the method. The average number of root categories per session is an indicator of how well the category vectors are able to calculate similarity across the branch boundaries of the category tree. Considering that the values are very similar across all comparison contexts and comparison methods (ranging from 1.21 to 1.24), it can be assumed that the majority of the similar categories per comparison method are within the same root category. These values are almost identical to the lexical baseline approaches. This means that having a similar category from another category branch is very rare. Looking at the data, again, this seems to be true; when calculating the number of sessions for the **u2vcac** approach with 1.24 root categories on average, only around 255,000 sessions can be found that have more than three root categories. Three or two root categories can be seen as common considering that, for example, the homepage comes with the **root_category_id** *1* (103,503,749 interactions in the dataset) and pages where it was not possible to assign a meaningful id have the value *42* (1,356,610 interactions in the dataset). It would seem that either a) there are no real similarities between different branches of the category tree, or b) the methods for choosing similar categories based on the cosine similarity are too strict, or c) the basic assumption for calculating the similarity was too naive.

The average number of categories per session displays the differences between the comparison methods very well. Although once again the variance is not very high (1.36–1.39 for the consecutive direct comparison and 1.41–1.46 for the interleaving approaches), it is clear that the different methods allow more or less similar categories on average. And in view of the fact that the cut-off sessions seem to provide the most similar categories for a given category, the average numbers here are the highest. Likewise, the numbers for the 0.5 threshold sessions are the lowest in comparison. The difference between the comparison context is smaller than expected.

The other measures look normal. The number of different products and the number of queries increases with the length of the session. They very slightly increase with the cutoff sessions, although the difference between the approaches is minimal. Apparently, the sessions with query interactions are similarly structured across the comparison methods.

	∅Time in session		∅Inter-Interactiontime		∅Interaction days	
	AM	SD	AM	SD	AM	SD
u2v05cd	3, 896.09	20, 701.56	1, 493.49	10, 174.5	1.23	0.77
u2v05cc	3, 985.66	21, 113.04	1, 498.02	10, 174.03	1.24	0.79
u2v05ad	7, 674.42	26, 073.1	2, 756.91	13, 330.88	1.33	0.92
u2v05ac	7, 965.07	26, 853.61	2, 771.59	13, 345.01	1.34	0.96
u2v10cd	3, 849.01	20, 440.93	1, 485.33	10, 131.28	1.23	0.76
u2v10cc	3, 984.4	20, 922.78	1, 506.34	10, 147.56	1.24	0.78
u2v10ad	7, 509.15	25, 611.62	2, 750.08	13, 289.76	1.32	0.9
u2v10ac	7, 921.96	26, 642.98	2, 771.74	13, 311.38	1.34	0.95
u2vccd	4, 209.07	21, 693.42	1, 604.54	10, 670.81	1.24	0.78
u2vccc	4, 356.17	22, 311.57	1, 611.81	10, 670.79	1.25	0.81
u2vcad	8, 150.85	27, 151.41	2, 925.81	13, 879.5	1.34	0.93
u2vcac	8, 642.22	28, 386.87	2, 950.2	13, 903.18	1.35	0.99

Table 5.29: User measures for u2v sessions regarding time spent. Abbreviations: AM = Arithmetic Mean; SD = Standard Deviation.

Table 5.29 shows the measures related to time spent on the site. As expected, the time in session and also the inter-interaction time is quite high and a lot higher than for all the mechanical approaches. They are lower than the lexical approaches and more or less comparable to the **bm25** sessions. The values are quite high overall. The time in session is usually around 2.5 to three days for the consecutive sessions and between five to six days for the interleaving sessions. The inter-interaction time ranges from around one to two days, indicating that many of these sessions span multiple days.

Interestingly, the average interaction days are relatively low. These values may have been disrupted by the number of bounces though. The maximum value for the **u2vac** sessions as the most strict one is 365 days, for example. This session has interactions on every day of the year. The other sessions – even the consecutive ones – have comparably high maximum values. When looking at the distribution of interaction days per session, the distribution clearly follows Zipf's law though – the majority of sessions have only one interaction day, no matter the session approach. Logically, the distribution is a lot smoother for users with a higher number of interactions, but it is still very clear.

Finally, Table A3 shows the top five sequences. The most important observation here is that the sequences are not that different from the previous approaches, even the mechanical ones, but the totals are lower than for the mechanical variants. At first glance, this seems surprising: not even the liberal logical sessions can connect certain interactions. On the other hand, as discussed above, these are likely to belong to all the sessions of users with a very low number of overall interactions as well as of users who looked only once into a certain topic or category. Furthermore, as can be deduced from the sessions consisting only of query interactions, these may be caused in addition by the preprocessing as explained in Section 4.4.3.2.

While incorrectly assigned categories to query interactions may strongly provoke session breaks for the consecutive sessions, apparently, they also influence the interleaving variants; as already observed, users sometimes only make partial use of the category tree, thus incorrectly assigned categories could lead to session breaks as well. Considering that the totals for the top five are lower in comparison, it can be safely assumed that the logical variants are more diverse and, most likely, longer than the mechanical variants.

Turning to look at the session breaks, similar values as noted before are reported: the user category vector variants all settle at around 14% interleaving sessions, with between two to 2.3 session breaks on average. There is no real difference between the comparison methods. Seeing that these approaches deliver numbers somewhat comparable to the **bm25** sessions, the similar percentage of interleaving behaviour makes sense. Apparently, interleaving behaviour is similarly identified across the session approach comparison method. This may well be worth investigating since it could reveal interesting details about user behaviour, although, naturally, the similar values could also be pure coincidence.

This concludes the section for the logical approaches based on category vectors and category similarities. While the numbers vary greatly from the mechanical approaches they do not so much from the **bm25** sessions. Structurally, they are not even that different; for example, there is a great overlap between the **bm25ac** and the **u2vcac** sessions with around 15m differently assigned sessions. As discussed above, there are some potential errors with this kind of approach which relates directly to data quality:

- tracetimes
- incorrectly assigned categories
- incorrectly used cut-off for similarity

These issues have to be kept in mind when implementing logical sessions – naturally, they are also valid for the other logical variants. Nevertheless, there is great potential, especially when aiming to understand highly engaged users. There are examples where the logical approaches connect sessions across the whole year, enabling an overview to understand the complete journey of a specific user in dealing with a specific topic.

For example, **user_ids** could be observed apparently using the system over the course of the whole year to renovate a bathroom, visiting different related categories from time to time with various other topics sprinkled in-between. Sure enough, a big part of the logical sessions finds similar sessions to the mechanical ones. Users with two overall interactions visiting product pages related to the same product in the same category will result in one session no matter the algorithm. The same is true for most of the users with a low number of overall interactions; if they visit the system once with a few interactions in the same category range, they will also get one session. If they visit the system throughout the year and land a certain interaction on a very specific category that they have not visited / will not visit again, this kind of behaviour will also result in exactly one session for the majority of approaches. The next section will combine the two types of session identification: it will be interesting to see how that changes the different measures.

5.2.3 Combined Approaches

In this section, the combined approaches are analysed. This type of ensemble approach mixes mechanical with logical components to combine both approach types and utilize their respective strengths. The section is structured like the previous one on logical session variants. All logical approaches are combined with a selection of the fixed temporal inactivity sessions. In addition, a widely acknowledged variant from the literature is tested and presented with the geometric sessions.

5.2.3.1 Lexical Baselines with Temporal Inactivity

In this section, lexical matching is combined with a fixed temporal inactivity threshold. First and foremost, combining the two components is intended to rid both approaches of their more prevalent weaknesses of never-ending lexical sequences and unrelated temporally close interaction sequences. Without any temporal boundary, lexical sessions have the potential to span the whole year. This may be reasonable for some systems, but will result in a very low number of overall sessions. Furthermore, a session lacking any such boundaries can hardly be used to measure short-term scale system performance. Mechanical sessions, on the other hand, tend to connect interactions that are actually unrelated. Combining the lexical sequences with an inactivity threshold may solve both problems to create shorter sessions focusing on one topic.

To see the effect of the combination, a selection of inactivity thresholds was combined with the lexical matching in both previously tested comparison contexts. Four values were chosen for the consecutive comparison: five minutes, 30 minutes (as the industry standard), 1,440 minutes and, for a broader scope, 14 days (i.e. 20,160 minutes). In addition, four values were chosen for the approaches allowing interleaving behaviour: 1,440 minutes, 14 days, 75 days and 180 days. Higher values for the interleaving sessions were imposed simply because a five-minute window (in which other related interactions could happen) was felt to be too low to make any significant difference in production. Thus, only values upwards of a whole day are tested. Likewise, larger windows seemed not to be very practical for the consecutive comparisons. It will be interesting to see how that assumption holds to reality.

	#Sessions	CV-R	B-R
lti5cdb1	428,209,378	58.36	42.1
lti30cdb1	373,299,068	66.95	39.07
lti1cdb1	303,701,559	82.29	34.05
lti14cdb1	246,895,650	101.22	30.5
lti1adb1	276,414,965	90.41	30.97
lti14adb1	195,715,922	127.69	25.99
lti75adb1	156,285,988	159.91	21.6
lti180adb1	144,934,232	172.44	19.7

Table 5.30: System measures for combined lexical sessions. Abbreviations: CV-R = Conversion Rate; B-R = Bounce Rate.

Table 5.30 lists the system numbers for the eight approaches. The list is ordered by comparison context and inactivity threshold. As expected to some extent, the differences are quite large. First, all consecutive approaches create more sessions than the lexical pendant without any temporal boundary. This is interesting because even with a 14-day window, the combined approach identifies around 17m more sessions than the purely lexical approach. This is an indicator that some users may very well work on the same topic for more than 14 days without any interruption at all. The interleaving approaches create more sessions as well – except for the variant with a 180-day window, which has 1m sessions less. Again, this can be explained by interactions with non-meaningful **category_ids** – while such events are only included in the original lexical variant if they are no more than 1,440 minutes from any other interaction with a meaningful **category_id**, the 180-day window variant includes these interactions when they are within a 180-day frame.

When looking at the individual variants more closely, it is interesting to see that the combined approaches all have more sessions than their mechanical counterparts as well. To some extent, this is expected but the differences are still quite high. This is an indicator of a different user behaviour; here, when comparing the number of sessions to the fixed-length or temporal inactivity sessions, the related sequences apparently get interrupted, starting a new session. This is especially visible for the variant with a five-minute inactivity threshold: the bounce rate of 42.1% indicates a lot of session breaks, with a noticeable increase when comparing it to **tf5** or **ti5**. Considering that the total number of identified sessions is very close to **tf5** though, the total number of bounced sessions will be higher. Nevertheless, similar observations can be made for the other consecutive sessions. The number of sessions and the bounce rate are considerably higher in comparison to the mechanical sessions, leading to the conclusion that the mechanical sessions deal with multiple topics whereas the combined approaches are now creating more shorter sessions.

The interleaving sessions are challenging to interpret because there are no real other comparable mechanical approaches. Placing them in relation to the original lexical variant, the number of sessions drastically decreases the lower the inactivity threshold is. This is very much expected behaviour for the 1,440-minute threshold, even though it is interesting to see that the purely mechanical consecutive sessions with a 1,440-minute threshold identify more sessions than the interleaving variant. Apparently, the time between sequences or interactions related to the same topic is eventually somewhat longer than one day, leading to session breaks here that may be connected by the purely mechanical variants. The divergences between the other variants are quite high as well; employing a 14-day threshold captures far more related interactions than using only a one-day threshold. The next gap is equally big – the 75 days again capture far more related interactions, leading to a drastically reduced number of sessions. Increasing the threshold to 180 days does not decrease the sessions as much, meaning that the majority of related interactions seem to happen within 75 days of each other.

Table 5.31 breaks down these numbers on a per-user level to bring additional insights. A first glance reveals that for the consecutive variants there are comparably high standard deviations for the average number of sessions. This is an indicator of differing user behaviour, where some users will visit a broad variety of topics where others only focus on a small subset. Equally, it could just mean that users work on a mixture of different topics rather than working on a set topic subsequently. Overall, the values here are logically higher than for mechanical counterparts as well as for the lexical variant. This is true for all measures. The measures behave accordingly for the interleaving variants too.

	⌀Sessions		⌀Interactions		CV-R		B-R		Lead-ins	
	AM	SD	AM	SD	AM	SD	AM	SD	AM	SD
lti5cdb1	5.46	17.68	3.37	3.02	63.58%	114.04%	36.0%	33.83%	1.36	1.04
lti30cdb1	4.76	14.71	4.01	3.91	79.41%	143.28%	30.24%	33.3%	1.58	1.21
lti1cdb1	3.88	9.99	4.67	4.88	93.28%	167.16%	24.44%	31.49%	1.85	1.54
lti14cdb1	3.15	8.12	5.46	6.67	108.79%	197.8%	20.11%	29.64%	2.2	2.41
lti1adb1	3.53	7.23	4.84	5.04	96.45%	170.06%	22.95%	30.7%	1.89	1.56
lti14adb1	2.5	3.1	5.98	7.94	118.3%	214.08%	18.13%	28.49%	2.37	2.75
lti75adb1	1.99	1.67	6.95	10.64	137.15%	257.81%	14.89%	26.57%	2.78	3.7
lti180adb1	1.85	1.32	7.33	11.68	144.7%	275.57%	13.65%	25.73%	2.95	4.06

Table 5.31: User measures for combined lexical sessions regarding system usage. Abbreviations: CV-R = Conversion Rate; B-R = Bounce Rate; AM = Arithmetic Mean; SD = Standard Deviation.

The bigger the inactivity threshold, the longer the sessions become, implicating increased values for the measures.

Overall, the approaches vary greatly although they all are based on the same comparison method and, to an extent, on the same comparison context. The inactivity threshold makes a big difference to how the resulting sessions are identified. It would seem that it is quite a challenge to capture user behaviour accurately; the correct balance between an inactivity threshold and the respective comparison context is quite elusive. Considering that, the basic assumption from the start of the section may not hold true; maybe the optimal approach will be an interleaving approach with a lower inactivity threshold.

	⌀Root Categories		⌀Categories		⌀Products		⌀Queries		⌀Topics	
	AM	SD	AM	SD	AM	SD	AM	SD	AM	SD
lti5cdb1	1.18	0.33	1.35	0.62	1.08	1.0	0.52	0.76	1.06	0.25
lti30cdb1	1.2	0.36	1.41	0.68	1.21	1.23	0.57	0.84	1.08	0.27
lti1cdb1	1.21	0.38	1.47	0.75	1.34	1.44	0.62	0.93	1.1	0.31
lti14cdb1	1.23	0.39	1.56	0.82	1.51	1.71	0.68	1.03	1.15	0.38
lti1adb1	1.22	0.38	1.49	0.77	1.37	1.47	0.64	0.98	1.11	0.32
lti14adb1	1.23	0.4	1.62	0.89	1.59	1.84	0.74	1.2	1.18	0.42
lti75adb1	1.25	0.41	1.78	1.07	1.81	2.23	0.83	1.52	1.28	0.58
lti180adb1	1.25	0.41	1.86	1.17	1.9	2.39	0.87	1.64	1.34	0.67

Table 5.32: User measures for combined lexical sessions regarding visited content. Abbreviations: AM = Arithmetic Mean; SD = Standard Deviation.

Table 5.32 shows the content measures. While the assumption might be that the values will be in close proximity, this does not correspond entirely with reality. Although the variance for the average categories per session is not that high with 1.35 to 1.86, the differences are still notable. Initially, one might assume that these values would be higher, considering that these sessions are connected based on the same root category; bearing in mind, however, that even the purely mechanical sessions had relatively low numbers here. It is still interesting to see that the purely mechanical variants have higher numbers on average; even comparable to the session approaches allowing interleaving behaviour. This is another clear indicator of user behaviour. Even the approach with a 180-day inactivity timeout has only slightly higher categories on average, meaning that most users interact with the system for a very limited number of categories per **root_category_id**.

Accordingly, the products and queries behave likewise. Interestingly, the number of average products per session is higher for the interleaving sessions compared to the purely mechanical ones, although the categories are on a comparable level. Potentially this indicates that the number of categories is not a crucial indicator of user behaviour, but the number of products is. Apparently, users visit more products in the same categories. This is a reasonable assumption overall, again considering the type of system and how users interact with it. The number of topics per session is also understandable. Logically, the higher the inactivity timeout, the higher the potential for multiple sessions.

	⌀Time in session		⌀Inter-Interactiontime		⌀Interaction days	
	AM	SD	AM	SD	AM	SD
lti5cdb1	1.63	10.47	1.13	11.02	1.0	0.01
lti30cdb1	4.23	12.23	2.02	11.02	1.0	0.02
lti1cdb1	85.63	263.34	34.76	115.03	1.06	0.21
lti14cdb1	1, 368.68	3, 964.88	436.02	1, 390.38	1.27	0.83
lti1adb1	93.17	276.39	35.9	115.96	1.06	0.22
lti14adb1	1, 819.36	4, 754.36	499.09	1, 455.14	1.34	1.01
lti75adb1	9, 932.04	23, 868.58	2, 520.99	7, 514.46	1.6	1.61
lti180adb1	19, 392.14	46, 603.31	4, 909.38	15, 667.83	1.71	1.83

Table 5.33: User measures for combined lexical sessions regarding time spent. Abbreviations: AM = Arithmetic Mean; SD = Standard Deviation.

Table 5.33 shows the time measures to underline the structural difference between these session approaches. Of course, the comparison context has the biggest impact, but the inactivity threshold seems to have a big impact as well, especially on the interleaving variants. Taking the differences seen here, these are likely to be caused by outliers; although the divergence between **lti75adb1** and **lti180adb1** seems too high considering the differences analysed previously. Other than this, the differences between the variants are relatively high.

Looking at the top five sequences in Table A3, the divergence between the consecutive variants is particularly apparent. The difference in the total percentages for the top five sequences is quite high; even the top sequences (which is as usual a single lead-in click on a product page) have a very different share between the different comparison contexts. The difference gets smaller the higher the inactivity threshold is.

The interleaving behaviour in these sessions is quite different. Logically, the number of sessions showing interleaving behaviour is dependent on the time constraint. For the long inactivity timeouts (75 and 180 days), the values are close to the baseline variant **ladb1**: 15.08%–17.36% with 2.73–2.85 session breaks on average. There is an interesting observation to make here in relation to the difference in the number of days: apparently, the gap between the 75 days and the 180 days is big enough to connect sessions using the longer inactivity timeout that actually belong together. For the smaller time constraints (one day and 14 days), the number of interleaving sessions as well as the average session breaks decrease: 4.87%–10.02% with 1.56–2.23 breaks. Despite having only a maximum of one day between interactions, there are still a relatively high number of interleaving sessions compared to the variants with a higher constraint; nonetheless, the overall numbers are quite low. The majority of sessions are straightforward with no real interleaving behaviour. This looks even more interesting when **lti1adb1** is compared with **lti1cdb1**: the divergence between the 276m–303m sessions is not explained solely by the number of interleaving sessions (about 13m sessions for **lti1adb1**), but caused, compellingly, apparently, by a small number of sessions with a very high number of session breaks.

5.2.3.2 Shared-Term Space with Temporal Inactivity

The next section combines the **bm25** similarity function with the fixed temporal inactivity thresholds. The basic assumption is the same as before: combining the strengths of both approach types to get rid of the weaknesses. The difference to the aforementioned lexical combination approaches are the additional comparison contexts as well as the different comparison method. It will be interesting to see whether the statements made previously about the self-containedness of the **bm25** category similarity is apparent here as well.

As before, similar values were chosen for the variants. For the consecutive comparisons, three values were chosen: five minutes, 30 minutes and 1,440 minutes. The variants with 14 days were omitted. For the interleaving variants, four versions were tested: 1,440 minutes, 14 days, 75 days and 180 days.

	#Sessions	CV-R	B-R
bm25ti5cd	462,413,589	54.05%	44.21%
bm25ti5cc	457,641,841	54.61%	44.31%
bm25ti30cd	412,472,229	60.59%	41.44%
bm25ti30cc	406,788,004	61.44%	41.59%
bm25ti1cd	353,801,526	70.64%	37.06%
bm25ti1cc	346,643,630	72.1%	37.28%
bm25ti1ad	319,821,010	78.14%	33.11%
bm25ti1ac	311,471,360	80.24%	33.42%
bm25ti14ad	258,863,980	96.54%	29.62%
bm25ti14ac	246,474,512	101.4%	30.31%
bm25ti75ad	235,749,939	106.01%	27.63%
bm25ti75ac	220,970,579	113.1%	28.5%
bm25ti180ad	229,260,341	109.01%	26.92%
bm25ti180ac	213,749,268	116.92%	27.83%

Table 5.34: System measures for combined bm25 sessions. Abbreviations: CV-R = Conversion Rate; B-R = Bounce Rate.

Table 5.34 shows the overall number of sessions and the associated conversion and bounce rates. Generally, the number of sessions is relatively high. Comparing the consecutive combined sessions to their purely mechanical counterparts, the combined sessions have far more sessions. As before, this is expected but the difference is very high. This may be interpreted as an indicator that the BM25 similarity is rather strict and even faulty to an extent. The bounce rate in particular is quite high across all consecutive sessions; the **bm25ti5cc** sessions with a five-minute inactivity threshold, shows over 200m bounced sessions. The sessions with a 1,440-minute timeout identify over 100m sessions more than their mechanical counterpart. Once again the increase in sessions is to be expected, but is rather high – especially considering that the bounce rate is also higher.

Although the interleaving variants identify fewer sessions, there are still far more than, for example, the lexical variants. Considering that the lexical variants had a comparatively low number of categories per session, it is somewhat strange to see the impact of the BM25 similarities quite so high. It seems that either the retrieved similar categories are off the mark of usual user behaviour or that they are highly specific, hence the higher number of session breaks, because they cannot possibly reflect usual user behaviour. It will be interesting to see whether the user-history category vectors perform differently here. Conversion rate and bounce rate perform rather uniformly, following the decreasing number of sessions. It is still very interesting to see the bounce rate this high, even for the interleaving variants with a 180-day inactivity timeout. A bounce rate of 26.92% for 229m sessions equals 61m bounced sessions. The sheer quantity of bounced sessions seems strange and is yet another indicator of the peculiarities associated with this type of similarity calculation.

The system measures shown in Table A6 follow the same patterns. Logically, there are fewer sessions overall with a higher inactivity timeout. The interleaving comparison contexts usually identify fewer sessions with more interactions: the sessions tend to be longer on average. The bounce rate on a per-user basis likewise stays high; on average around 20% of all sessions per user, even for the 180-day window. This could be down simply to somewhat unconnectable categories, but a more likely explanation is that low-interaction users looking at unrelated categories are disturbing the average here.

The content measures in Table A7 are quite similar in-between the approaches. It is interesting to see that the number of average categories per session remains similar to the lexical approaches, despite the assumption that the BM25 similarity calculation is a lot stricter. The deep dive into the data that would be necessary to uncover the issues here are outside the scope of this dissertation. It is somewhat questionable to see similar average content- and time values along with far more sessions. Nevertheless, the choice of comparison method, context or inactivity threshold does seem to make an impact, which may explain the quite big differences between all these sessions. The effect is quite similar to the lexical combination approaches. When the time-related measures – which can be found in Table A8 – are compared to the combined approaches using lexical matching, the values are lower. This is just as expected and explains the higher number of sessions. The differences caused by the temporal threshold are also naturally very high between the approaches.

The fact that the introduction of the temporal threshold raises the potential of more session breaks in the **bm25** sessions is a logical effect. Still, the increase when introducing an inactivity timeout of five or 30 minutes is quite high; 100m more sessions, prompted only by the implementation of a 30-minute timeout is a significant divergence, which causes very different system measures. Interestingly, the content measures do not differ that much. Ultimately, the tendency of the **bm25** sessions to impose session breaks seems greater than for the other logical approaches. The similarity retrieval algorithm works, but needs more fine-tuning: for example, the content of the e-commerce-specific text may not be suitable for finding similar categories without further adjustment to the retrieval algorithm (or the documents, in any case).

5.2.3.3 Category Embeddings with Temporal Inactivity

This section analyses the logical approaches in combination with a mechanical timeout. The category similarity determined by word2vec on the user history is used in combination with a temporal inactivity timeout. The intention is the same: to allow logical sessions a mechanical ending to potentially have no sessions that span the whole year.

	#Sessions	CV-R	B-R
u2v05ti5cd	457,028,558	54.68%	44.39%
u2v10ti5cd	455,774,632	54.83%	44.03%
u2vcti5cd	450,926,682	55.42%	43.76%
u2v05ti5cc	455,991,403	54.81%	44.4%
u2v10ti5cc	453,378,131	55.12%	44.08%
u2vcti5cc	448,845,900	55.68%	43.8%
u2v05ti30cd	406,679,566	61.45%	41.74%
u2v10ti30cd	405,342,083	61.66%	41.31%
u2vcti30cd	399,878,149	62.5%	41.0%
u2v05ti30cc	405,332,467	61.66%	41.75%
u2v10ti30cc	402,418,159	62.1%	41.38%
u2vcti30cc	397,310,961	62.9%	41.05%
u2v05ti1cd	347,716,468	71.87%	37.5%
u2v10ti1cd	346,305,355	72.17%	36.95%
u2vcti1cd	339,887,635	73.53%	36.55%
u2v05ti1cc	345,822,886	72.27%	37.52%
u2v10ti1cc	342,511,906	72.97%	37.04%
u2vcti1cc	336,459,169	74.28%	36.63%
u2v05ti14cd	308,368,930	81.04%	35.34%
u2v10ti14cd	306,960,624	81.42%	34.68%
u2vcti14cd	299,303,431	83.5%	34.19%
u2v05ti14cc	305,770,681	81.73%	35.38%
u2v10ti14cc	302,146,443	82.71%	34.83%
u2vcti14cc	294,754,777	84.79%	34.31%

Table 5.35: System measures for consecutive combined u2v sessions. Abbreviations: CV-R = Conversion Rate; B-R = Bounce Rate.

The structure is a bit different now, since the number of varying session approaches is far bigger due to there being three separate comparison methods: top 10 most similar categories; 0.5 threshold; and cut-off sessions. All these methods are tested with the same set of temporal inactivity thresholds: five minutes, 30 minutes, 1,440 minutes, 14 days for the consecutive variants, 1,440 minutes, 14 days, 75 days and 180 days for the interleaving variants. In combination with the comparison contexts, that makes 48 different variants, 12 combinations for every comparison method variant.

Table 5.35 shows the overall sessions for all variants relying on consecutive comparison. The list is ordered by comparison method and comparison method and temporal inactivity threshold. There are three rows per comparison context and threshold, representing the different comparison method subvariants. The differences are quite high when the complete list is considered, but are not actually that high when looking at the differences in the individual subvariants.

The overall variance ranges from 457m to 294m sessions when comparing the approaches using a five-minute timeout with the variant of a 14-day inactivity gap. Looking at the comparison method subvariants and the comparison context subvariants, the differences are not that high when staying in the same comparison category. The biggest difference is induced by the inactivity timeout. This can also be seen in the bounce rate, which decreases with a higher timeout. The variance here is not that high – roughly 10% difference from 44.39% to 34.19%, but the difference in absolute numbers is logically higher (decrease in bounced sessions 202m–102m with a difference of around 158m sessions overall). Again, this is caused by the timeout rather than the comparison method.

This is the most interesting observation here. The differences between the comparison method subvariants are very small again, even smaller than for the purely logical approaches. As before, the calculated cut-off seems to include the most similar categories per category, while the 0.5 threshold and top 10 most similar category variants are somewhat close to each other, delivering mixed results – sometimes, there are more sessions when using the 0.5 similarity, other times there are more with the top 10 most similar categories. This is apparently dependent on the threshold and comparison context as well, at least to some extent. This could mean that some vaguely similar categories are visited less often than others – or only after a more extended time frame. Since the effect is so small (the differences are usually around 2m sessions), this could also be down to anomalies in the data or in the similarity calculation. The biggest effect on the differing session numbers seems to be caused by the timeout. The lower that is, the more restrictive the session models work, introducing more potential session breaks by cutting short engagement in topics.

Table 5.36 shows the overall system values for the interleaving sessions in the same way the consecutive variants were displayed. The same effects are visible here. The overall variance is huge when looking at the overall totals: a decrease of 310m to only 203m sessions. Likewise, the bounce rate decreases from 33.35% (103m bounced sessions) to 26.86% (56m bounced sessions). Again, the differences are not that big between the comparison method subvariants and comparison context subvariants; here, the same behaviour as seen in the consecutive sessions can be observed. Clearly, the actual number of similar categories per reference category makes a difference, but in the case of the chosen variants the differences are small. Here, there is no notable difference to the purely logical approaches using the category vectors. There is a somewhat notable difference in comparison to the **bm25ti** sessions, however; here, the impact seems to be slightly greater. The overall effect is as anticipated and as already observed with the other combined approaches. Generally, the number of sessions increases when introducing inactivity thresholds to the logical comparisons, except that the non-meaningful categories are easier to connect with the higher thresholds. Were that the case and they were still treated differently, the number of sessions would be higher still. Again, it is interesting to note that the higher time thresholds seem to have little impact on the interleaving comparison contexts. In addition, once again, the differ-

	#Sessions	CV-R	B-R
u2v05ti1ad	310,045,159	80.61%	33.35%
u2v10ti1ad	311,566,308	80.21%	33.06%
u2vcti1ad	306,540,785	81.53%	32.82%
u2v05ti1ac	308,130,602	81.11%	33.41%
u2v10ti1ac	307,315,533	81.32%	33.22%
u2vcti1ac	302,831,132	82.53%	32.95%
u2v05ti14ad	247,383,439	101.02%	30.28%
u2v10ti14ad	249,713,150	100.08%	29.8%
u2vcti14ad	243,611,312	102.59%	29.48%
u2v05ti14ac	243,850,987	102.49%	30.43%
u2v10ti14ac	242,926,971	102.88%	30.18%
u2vcti14ac	237,279,638	105.33%	29.81%
u2v05ti75ad	223,642,592	111.75%	28.58%
u2v10ti75ad	226,320,634	110.43%	27.99%
u2vcti75ad	219,474,949	113.87%	27.56%
u2v05ti75ac	218,928,488	114.15%	28.8%
u2v10ti75ac	217,898,455	114.69%	28.47%
u2vcti75ac	211,303,578	118.27%	28.0%
u2v05ti180ad	217,009,417	115.16%	27.96%
u2v10ti180ad	219,750,212	113.73%	27.33%
u2vcti180ad	212,709,792	117.49%	26.86%
u2v05ti180ac	211,920,760	117.93%	28.19%
u2v10ti180ac	210,812,680	118.55%	27.84%
u2vcti180ac	203,974,303	122.52%	27.32%

Table 5.36: System measures for interleaving combined u2v sessions. Abbreviations: CV-R = Conversion Rate; B-R = Bounce Rate.

ences between 75 days and 180 is marginal no matter the comparison method subvariant or comparison context subvariant. In the case of the consecutive approaches, the time threshold makes a much larger difference no matter how big it is. Again, the comparison method subvariant makes only a subtle difference here.

The system measures shown in Table A9 on a per-user basis are hardly surprising. They follow the same patterns as the overall system measures, where the average numbers of sessions and interactions per session are closely related to the overall number of sessions. The same is true for the conversion and bounce rates and logically also for the lead-ins. It can be confirmed that the divergences, mostly introduced by the different inactivity thresholds, are large. As can be seen in Table A10, from a content perspective, the average numbers are close to the logical approaches. Similarly, commonly there is only one root category per session and on average 1.26 to 1.46 categories per session, which is a slightly lower number than the typical mechanical session. Again, the number of products and queries is related to the number of identified sessions, which decrease with a higher inactivity threshold. As expected the identified topics are very low, as are the differences in logical approaches. This could mean that either the sessions are still very similarly structured or that the measures are somewhat distorted by the quantity of data – both reasons are likely but considering that the standard deviations are also at least somewhat similar, the first assumption seems more likely.

The time measures in Table A11 are also directly affected by the mechanical thresholds. Naturally, the average time in session and the inter-interaction time are usually lower than for the mechanical pendants, this is because the logical comparison between events introduces another dimension that potentially leads to shorter sessions. The same is true when comparing them to the logical pendants without any mechanical boundary; the variants with 180 days are very close though. This is again only logical because a 180-day inactivity timeout will only rarely have an effect.

5.2.3.4 Geometric Approach to Session Segmentation

This final section dealing with the combined approaches looks at the geometric session-identification approach introduced by Gayo-Avello in 2009 [80]. This session approach was developed to incorporate both temporal and lexical distance between query interactions, eventually combining them into a single distance function utilized to decide on session break or session continuation. Originally, the lexical distance was computed by matching query terms by way of character n-gram comparisons. This dissertation uses the cosine similarity calculated previously to measure the similarity between category vectors based on user history. This choice was made because the data lacked enough meaningful queries and because the session identification aims to connect contextually related pages, which are identified by the **category_id** that is inherent to each page.

The algorithm usually takes a value range between 0 and 1 as input. Since this would theoretically include all of the over 2,300 categories for a given category, the lower boundary was set to be 0.1 to achieve similarities that are more meaningful. While this helps in the capture of only vaguely similar categories to use for the comparison, it may still be a) either too liberal or b) too strict, as the discoveries made in the previous sections have outlined. Thus, while the algorithm works, this may nonetheless be a good point to fine-tune it to achieve better quality results.

The values for the temporal distance are also represented in the range from 0 to 1. The temporal distance might be seen as a degrading factor that decreases as more time passes between two interactions. For the experiments, three values were chosen: one day, 14 days and 75 days. Smaller values are not reasonable because of the degrading nature of the temporal distance. For example, with a maximum distance of 24 hours (i.e. one day), the category similarity must be very high for interactions that are around 24 hours apart to be able to connect these two interactions into one session. While the original algorithm tested only the 24-hour window as a reasonable time period for user interactions with a system, this dissertation experiments with higher values as well, to see if the algorithm is sufficiently robust to work with these numbers. Thus, the assumption here is that a higher temporal distance could make the comparison too liberal.

All four comparison contexts were tested. The original work by Gayo-Avello [80] only tested consecutive comparison with a reference event and the complete session history. Here, the assumption is as usual; the comparison context will have a big impact on the session numbers because the interleaving behaviour will circumvent potential session breaks.

	#Sessions	CV-R	B-R
geomu24cd	296,104,692	84.4%	32.52%
geomu24cc	290,945,430	85.9%	32.61%
geomu14cd	247,540,067	100.96%	29.67%
geomu14cc	238,492,729	104.79%	29.92%
geomu24ad	280,213,676	89.19%	30.7%
geomu24ac	275,590,269	90.68%	30.82%
geomu14ad	218,814,074	114.21%	27.29%
geomu14ac	210,212,581	118.89%	27.7%
geomu75ad	192,194,826	130.03%	25.09%
geomu75ac	182,000,121	137.32%	25.58%

Table 5.37: System measures for geometric sessions. Abbreviations: CV-R = Conversion Rate; B-R = Bounce Rate.

Table 5.37 shows the high-level numbers for the geometric sessions. The variance overall is around 114m sessions when comparing all variants. The differences between the comparison context subvariants are quite small but between the overarching comparison context the differences are higher. This is again logical. Comparing the identified sessions to the combined variants from the previous section, the conclusion is that the initial assumption is true: the temporal value is very liberal due to its degrading, and the geometric approach identifies far fewer sessions. The effect observed here is that temporally close interactions are connected even when the similarity score of the categories is low.

Interestingly, the bounce rate remains highly comparable with the other approaches. Even though the degrading temporal inactivity should cause even loosely connected interactions to be combined into one session, there is still a 25% share of bounced sessions even for the 75-day variant. These are still 45m sessions with only one interaction. This might be caused by the 0.1 gap introduced for the category similarity, whereby even very close interactions will not be connected if their similarity is below 0.1. This limitation may be a bit too harsh, but otherwise larger temporal factors are likely to connect close interactions no matter what. This would be a good point to fine-tune these approaches since

	⌀Root Categories		⌀Categories		⌀Products		⌀Queries		⌀Topics	
	AM	SD	AM	SD	AM	SD	AM	SD	AM	SD
geomu24cd	1.27	0.46	1.51	0.79	1.36	1.44	0.65	0.98	1.11	0.33
geomu24cc	1.28	0.47	1.53	0.84	1.37	1.48	0.66	1.03	1.12	0.37
geomu14cd	1.3	0.48	1.57	0.85	1.48	1.65	0.7	1.06	1.14	0.37
geomu14cc	1.31	0.5	1.6	0.94	1.51	1.93	0.72	1.45	1.16	0.43
geomu24ac	1.28	0.48	1.54	0.85	1.39	1.5	0.67	1.06	1.13	0.37
geomu24ad	1.28	0.46	1.52	0.8	1.37	1.45	0.66	1.0	1.12	0.33
geomu14ad	1.3	0.48	1.59	0.87	1.51	1.69	0.72	1.11	1.15	0.38
geomu14ac	1.31	0.5	1.62	1.0	1.56	2.18	0.76	1.94	1.17	0.45
geomu75ad	1.32	0.5	1.65	0.92	1.62	1.86	0.77	1.2	1.18	0.44
geomu75ac	1.34	0.53	1.7	1.17	1.7	2.72	0.83	2.73	1.21	0.57

Table 5.38: User measures for geometric sessions regarding visited content. Abbreviations: AM = Arithmetic Mean; SD = Standard Deviation.

they are directly dependent on the similarity comparison. In the use case here, connecting interactions of categories that are only very vaguely similar was not intended.

The system measures differ accordingly as shown in Table A12, with the differences following the overall system numbers. The average number of sessions ranges from 3.78 to 2.32 with average interactions per session from 4.78 to 6.64. Per user, the bounce rate is also lower, being reduced to a share from 23% to 16.5%. Potentially, these bounces could have been avoided had an even bigger time period been imposed or the 0.1 gap been removed. Nonetheless, it is important to remember that the similarity should be handled with care, since it has been shown that rules that are too-liberally imposed with a high temporal inactivity result in a very low number of sessions. Even now, the average number of sessions is already relatively low. The majority of users are likely to have only one session in this constellation.

Table 5.38 shows content measures following the assumptions already made. The fact that there is a slightly higher number of root categories and categories per session is a clear indicator that nearly the full range of category similarities have been utilized. The higher number of average topics per session underlines this: the geometric approach usually connects more categories and therefore more topics in one session than all the other approaches. Whether this is a good or a bad thing is a question to ask from a qualitative perspective; the question is deserving of close analysis before putting such an approach into a production environment. Then again, this is the case for all approaches that use the calculated similarity, because they all depend on the quality of these similarity comparisons.

From a time perspective, there are huge differences between the different variants as shown in Table A13. Naturally, this is expected, but still the variance is relatively high. The same behaviour has been observed for the other approaches combining mechanical and logical components. The values seen here are slightly smaller in comparison to the user category vector approaches, for example, but this is reasonable considering that more interactions are usually connected, creating smaller gaps.

This concludes the summary for the geometric session approach. The actual concept seems to work quite well, but more fine-tuning of the similarity between interactions is required. As the distance measures used to compare interactions cross-influence each other,

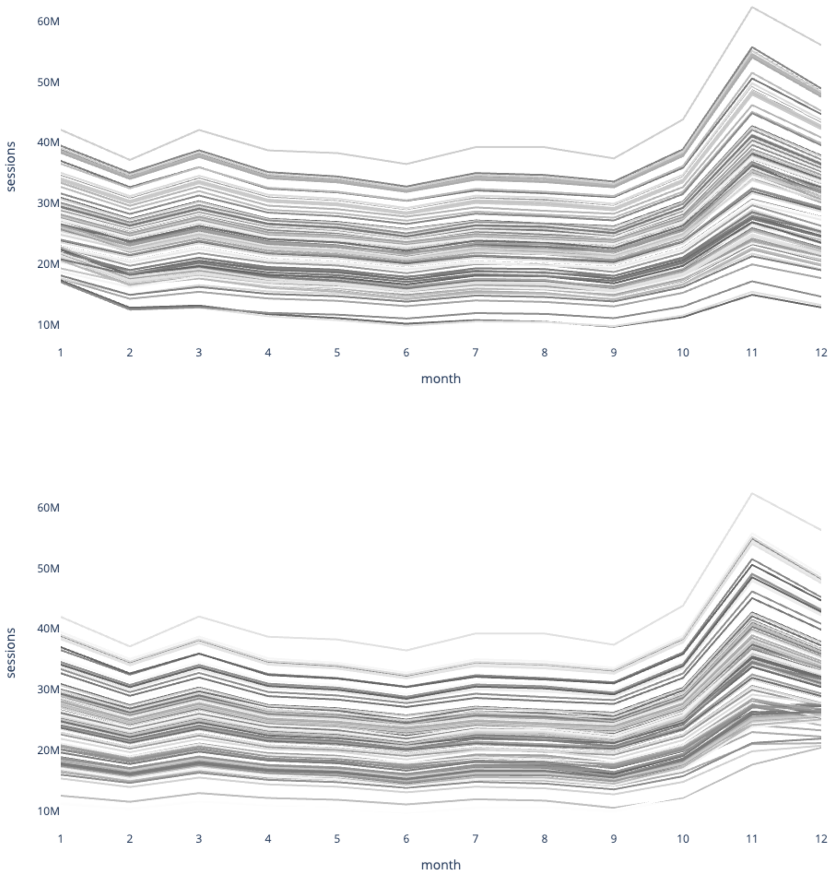


Figure 5.15: Line chart of all session approaches with their respective number of identified sessions over time: per month of their starting day (top) and end day (bottom).

they need careful evaluation and fine-tuning to work optimally. The similarity between categories may be harder to balance than similarity in natural language – it may well be the case that the algorithm is not fully suited to working with this kind of data.

5.3 Discussion

This final section discusses the previously analysed results of all session approaches. Although many points have already been discussed in the context of the various approaches, this section will briefly summarize these again and highlight the most important points, bringing to the fore the most notable differences between the various approaches. Finally, a selection from all the approaches is made in preparation for the next chapter.

Before a description of the advantages and disadvantages, Figure 5.15 shows all session approaches again over the course of the year. The figure at the top shows the number of sessions per starting day – meaning the day of the first event of a session – while the bottom figure shows the number of sessions per end day – the last interaction of a session.

The intention of the two figures is to map the session behaviour on the system over the course of the year to observe any general structural differences.

The differences in the numbers of sessions are very clear. The more interesting observation is the close similarity of the trends between the 135 session approaches. When looking at the sessions per the month of the session start day, the various approaches behave very much alike. The lines of session approaches with a higher number of sessions (i.e. the combined approaches with a five-minute inactivity timeout) show steeper trends than the approaches with fewer overall sessions, but the general trend is very similar. January represents a special case here: where usually the session approaches identify sessions as relatively stable throughout the year (except for November due to Black Friday), the approaches in the lower third of the top figure identify the most sessions in January (or at least a comparable amount to November). The reason for this is the longevity of the identified sessions by logical or combined approaches with a higher timeout; they may span the whole year if the user comes back time and again to work on the same topic.

This observation is underlined by the bottom figure, displaying sessions by their end day, meaning the day of the last interaction of a session. Here, the general trends again are very similar; the line chart actually closely resembles the one showing sessions by start day. This is expected to some extent; it signals that the majority of sessions start and end on the same day, even the logical and combined approaches. As seen particularly well in the bottom figure, a high number of sessions end in December (even more than in November), contrary to the other approaches with more identified sessions. These session approaches identify sessions that span the whole year or a long period of time, a bigger proportion of them ending in the last month or even on the last day of the year. Presumably, were new data fed into the algorithms, these sessions would go on for even longer.

Table A14 lists all approaches again in direct comparison. The table shows the overall numbers of sessions, the bounce rate as well as the average root categories, categories and products per session. As can be seen, the differences between all approaches is fairly high. The variance in the number of sessions ranges from 513m to 144m, while the bounce rate follows accordingly – 51.5% as the highest share of visits and only 19.7% for the **lti180adb1** sessions – the approach with the fewest number of sessions. The table is ordered by the number of sessions. Visually, it can be roughly divided into three parts: the first third is made up of the combined approaches with a very small temporal threshold, followed by multiple mechanical sessions and some combined / logical approach in-between and almost exclusively combined approaches in the bottom third.

The structure is a logical one: a combination of the more or less strict logical comparison method and the additional low mechanical boundary identifies a relatively high number of sessions with a high bounce rate. All the combined approaches with a five-minute boundary create more sessions with a higher bounce rate than the **tf5** sessions; this observation is interesting, since the **tf** sessions simply split interaction sequences into temporal buckets. The fact that these five-minute buckets identify fewer sessions than the combined approaches indicates that the combined approaches are probably too short. The increased bounce rate is another indicator of that. The combined approaches with a 30-

minute timeout now follow. These identify slightly fewer sessions but still have a bounce rate of above 40%. The variance here, however is not particularly high.

A big block of mechanical sessions then follows this, with the occasional combined approach in-between. It is noteworthy that even the **ti30** sessions with the 30-minute industry standard inactivity threshold identify fewer sessions than the combined approaches with a one-day temporal inactivity threshold and the consecutive comparison contexts. As discussed above, the **td** sessions come very close in terms of numbers to the **ti30** sessions with a noticeable overlap. Likewise, there is a high overlap between the **tf** and **ti** sessions. Most of the interactions for the majority of the user base seem to happen in very close proximity.

Many more logical approaches along with combined approaches follow in the bottom section of the table that show fewer identified sessions than the mechanical variants. The last third of the table is almost entirely comprised of the combined approaches with a high temporal boundary and the interleaving comparison contexts – mixed with the logical approaches with no temporal boundary at all. The smallest number of sessions is identified by the baseline approaches that use simple matching between root categories and the geometric approach that uses the calculated similarity from the user category vectors in a quite liberal way. Logically, these approaches have also the lowest bounce rates – they simply connect interactions very freely.

More interesting insight is found in the differences in averages of visited root category and category per session, as well as the products. The measures in the table are calculated across all sessions without aggregating at **user_id** level beforehand. The average number of root categories is consistently very low across all session approaches. The highest values are reached by the mechanical approaches with the higher inactivity or fixed thresholds with around 1.3 visited root categories per session. Only the more liberal combined approaches, the geometric variants with the higher timeouts, can reach the same level. Overall, the differences are relatively small though, which is somewhat expected as the number of root categories is limited. The majority of sessions will only visit one root category (not counting the pages with a non-meaningful root category).

This changes when looking at the actual visited categories. There are rather noticeable differences between the different approaches. The highest value, with 2.03 categories per session, is achieved by the baseline approaches with lexical matching, in combination with inactivity as well as purely logical. This is reasonable considering their nature and comparison method. Aside from that, the values are relatively evenly distributed at around 1.5 categories per session. Most interestingly, there is not that much difference between logical and mechanical sessions. This is especially visible for the combined variants with a low inactivity threshold and their mechanical counterparts: while **ti5** and **tf5** have 1.38 and 1.35 categories on average, the ensemble pendants only have between 1.22 and 1.28. Even the wider ranging one-day combined variants have only around 1.3 categories. The variants responsible for interleaving behaviour only have slightly higher values: from 1.42 categories for **u2vcti14ac** to 1.51 for **u2vcti180ac**. The geometric variants have around

1.65 categories for the 75-day inactivity approaches. The consecutive variants usually have a lower value.

The purely logical variants all have between 1.3 and 1.5 categories (except for the aforementioned baseline variants). There is usually a visible difference between consecutive and interleaving approaches. When comparing the consecutive logical variants to the mechanical approaches, the mechanical variants usually have more categories; especially for the longer inactivity timeouts. The latter also have comparable numbers to the non-consecutive logical variants. The same observations can be made for the products, although the values are usually lower. The values appear to be slightly higher for the non-consecutive combined approaches in comparison to all other approaches.

Advantages and Disadvantages of Approaches

These point to the main differences between mechanical, logical and combined approaches immediately apparent here. Mechanical approaches do not usually span longer time periods, whereas purely logical or combined approaches with a longer threshold tend to go on for a very long time – at least among the more engaged users with a higher number of interactions. Among users with a smaller number of interactions, a very similar number of sessions is usually seen no matter the session approach (though logically this depends on how users interact with the system). Ultimately, all these possibilities offer advantages and disadvantages. Following a user journey over the course of a year in, for example, renovating a bathroom is valuable information; likewise, it may be interesting to know that this user visits the system regularly for a set period of time and works on different topics in a sequence with limited interruption.

A discussion of the advantages and disadvantages may lead to the conclusion that different session approaches identify sessions of differing extents and in search of various types of information. Finding the right approach for a system or the right set of approaches is most likely the key to be able to fully understand the user population. Of the 135 session approaches analysed in this dissertation, some have definitely performed better than others in the sense that the identified sessions have come closer to reasonably representing user behaviour. In other words, identifying these sessions under the assumption that a user performs some interactions within a 30-minute time period, in three sessions using a five-minute inactivity timeout, is probably unreasonable (although, of course, this is debatable). Data quality is an additional factor that has direct implications for the performance of these algorithms and, therefore, on the quality of the sessions. With this in mind, some approaches can be discarded right away; these will not be considered in the second part of the evaluation and utilizing them in a production environment is not recommended.

Visits: Path-Based

First and foremost, these data-quality issues show up in the visits. In theory, the path-based approach is the closest representation of user behaviour. When a user enters the system and clicks through various pages, meanwhile opening the system in a new tab looking for different (or the same) topics, only the path-based approach is able to actually catch two different sessions here[4]. Moreover, it is only by connecting the different interactions using the path a user takes that it is possible to trace user movement – the visits reveal which pages are visited first and in which order certain pages or actions are called. This is not necessarily based on temporal order and can only be retrieved by looking at the relationship between **url** and **http_referer**.

This might be valuable information when analysing user behaviour: to learn whether users get stuck on a certain page, or which pages are visited before generating income and fulfilling business goals, may be of great help to improve a system. The same can certainly be achieved to an extent with mechanical or logical approaches, but not as concisely and specifically as the path-based approach. The reliance of the path-based approach on clean and specific information is simultaneously its biggest weakness and even risk. As the analysis has already discussed, the visits are entirely dependent on the quality of the data.

With 58m sessions with no clear origin and the highest bounce rate of all approaches, the path-based concept's data quality seems not good enough for it to work in practice. There are various examples where the algorithm is not able to connect interactions although they belong together; either because of marketing campaign parameters in the **http_referer** that are not transferred to the **url** of the next interaction or reloaded content that removes the connection between two interactions. What is more, the untracked parts of the system have the potential to lead to recurring session breaks. The high number of identified sessions suggest the same. Overall, the visits are too dependent on clean and functional tracking to be able to work flawlessly, and the information gained by correctly identified visits could be superseded by too many incorrect assumptions. Thus, for the purposes of this dissertation, as the visits were deemed to be too flawed, they are not used further in the evaluation.

Mechanical Sessions

The mechanical session-identification approaches seem to deliver stable results. They are easy to understand, easy to implement and can be considered a reliable way to track user engagement with a system over time. The different variants – fixed session length, fixed inactivity timeout, dynamic inactivity timeout – behave surprisingly similarly, indicating certain global patterns in user behaviour. Again, identifying these patterns may be very helpful in improving a system. There are various easily spotted examples in the data.

For example, some interesting insights have been revealed about how the different timeouts lengths connect with the number of identified sessions. The most prominent example is the obvious overlap between the **ti30**, **ti45** and all the dynamically calculated timeout sessions. The statistics relating to the dynamic timeouts at first appear as if they would offer a more diverse landscape of different session lengths depending on the property used to calculate them, but ultimately, the results are almost identical (within the **td** sessions) and in-between the **ti30** and **ti45** sessions with a relatively high overlap. This indicates a global timeout somewhere in that timeout area – not necessarily the 30 minutes, but somewhere close.

[4] Technically, the logical approaches could do the same job, but not as cleanly as the path-based approach.

This is underlined by the fact that the information gain (i.e. the number of identified sessions) decreases very little with a longer timeout frame. The difference in identified sessions between **ti30** and **ti45** or **ti60** is rather low compared to the increase of the actual timeout. This logically changes when using 1,440 minutes as a timeout because this approach connects sessions on different days; the gain is still not that high though. Likewise, the strong overlap between the **tf** and **ti** sessions is another argument here. There is only a minor difference between the **tf30** or **ti30** sessions, indicating that the inactivity timeout is seldomly applied at all; all interactions happen within the 30 minutes from the first interaction anyway. The same is true for the many other **tf** sessions and their **ti** pendants.

Other obvious examples of this are the session approaches with a 720-minute timeout or a 720-minute maximum session length, as well as the approach that simply combines all the interactions of a user per day into one session. These three variants identified almost exactly the same number of sessions with a variance of only 4m when comparing **tf720** and **ti720**. That these approaches have 60m more sessions than, for example, the 30-minute variants suggests that two sessions per day are hardly a rarity (at least when 30 minutes is considered the standard session).

From a content perspective, the mechanical sessions are also very similar, which is logical considering the overlap. The averages for categories and lead-ins per session for these session variants is somewhere between 1.3 to 2.1 for the lead-ins and 1.47 to 1.65 for the categories. The lead-ins indicate that these sessions definitely do not represent user behaviour as accurately as the visits would – unrelated user visits are connected here. The average number of categories, on the other hand, seems pretty plausible when taking a view across all session approaches.

At any rate, the mechanical sessions seem to be very good at capturing regularly reoccurring user behaviour. Depending on which factor is significant – behaviour on the system or regular activity – the timeout can be adjusted to represent the needs of the system. The lower the timeout, the more granular the representation of user behaviour becomes. In relation to the number of identified sessions, a timeout of between 30 to 90 minutes seems reasonable for a standard session, while the 720-minute timeout seems simply to capture session days – just like the session-day approach. As the lower timeouts and 1,440-minute timeout sessions tend to split user behaviour differently these are seen as special cases. Extended time ranges would probably be more reasonable to better capture user behaviour over certain time periods, while the shorter timeouts are probably simply unreasonable choices.

Logical Sessions

The logical approaches represent the big black box of the session approaches in this dissertation. Where the mechanical sessions are predictable in their way of comparing interactions and identifying sessions, the lexical and semantical comparisons are not. The comparison of interactions based on their topical relatedness instead of just subtracting timestamps was an experiment conducted without any expected outcome. The introduction of comparison contexts added another dimension and yet another level of complexity.

The general observation is that there are divergences between the different comparison methods, but not as strong as one would expect. This could signal a good working category-similarity system. The difference between the user category vector approaches has been discussed above; the number of identified sessions is also somewhat close to the number of **bm25** variants. The comparison method or rather the system to calculate similarity does not seem to be overly important – at least when only the user category vector variants are being compared. The thresholds require fine-tuning, but the differences between these are minimal. Likewise, in comparison with the **bm25** sessions, there are minimal differences, although the identified sessions could be seen to differ greatly. Looking at the data again, indeed an overlap exists between **bm25ac** and **u2v05ac**, showing only a rather small number of differences.

Only the lexical matching baseline variants differ significantly, identifying fewer sessions than all the other logical variants. This is reasonable as discussed previously. Connecting all sessions of a root category branch leads to very broadly scoped sessions. Of great interest is the consecutive baseline: with 229m identified sessions, this approach still identifies less sessions than, for example, the **tfd** session per day approach, indicating that users will work on the same topic over multiple days, or rather work in multiple sessions, without interruption.

A general observation could be that there is no real need for the fine-grained comparison contexts: the difference in outcomes between, for example, the **ac** and **ad** sessions is in most cases negligible. There are differences (usually around 10m sessions for the interleaving contexts, even less for the consecutive variants), but the difference overall is so low that the **ac** and **cc** (or **ad** and **cd**) sessions may be considered good enough. This may hold true only for the use case in this dissertation though. Generally, it seems that the comparison context is not that important for the purely logical approaches.

Again, the most important aspect is data quality: having recalculated the approaches, the number of identified sessions vary from run to run. As discussed, this is caused by identical timestamps of interactions of the same user, leading to non-deterministic behaviour. But incorrectly assigned or even missing categories can also lead to problems. Especially the preprocessing step of assigning categories to queries may have introduced errors that need to be handled when using logical sessions in a production environment. Altogether, while the resulting errors or problems may not be that critical they certainly do make an impact on the resulting measures, thus potentially leading to incorrect interpretations and decisions. The influence of data quality is certainly higher here than for the mechanical approaches.

Ultimately, while there are some highly significant differences between the logical sessions and mechanical variants, at the same time they do not differ that much. Looking at the consecutive approaches, these have structural similarities to the mechanical approaches. They can be seen as finer-grained parts of the mechanical sessions, dividing the sequence of interactions of a user into topically related parts. Following this assumption, these approaches could be used to do just that: mechanical sessions could be used to observe user engagement over time while the consecutive logical approaches could be used to see how users approach certain topics. The same is true for the approaches allowing interleaving behaviour: these enable system owners to understand users' engagement with a certain topic or even a collection of related topics over time, improving understanding of the time users take to fulfil a business goal.

Combined Approaches

Finally are the combined approaches, which attempt to combine the properties of both the mechanical and the logical session-identification approaches. This can be directly observed in the numbers of the identified sessions as well as in the properties of these sessions. Overall, the combined approaches tend to identify far more sessions than their mechanical counterparts and a comparable quantity to the logical variants, at least with the higher timeouts.

Again, the comparison context subvariant does not seem to be that important here. The combined variants within the same context variant identify a comparable number of sessions with negligible differences. The inactivity timeout seems to be the most important factor. While the comparison context and the comparison method seem to only marginally change the resulting sessions, the timeout drastically increases or decreases the outcome.

To weigh up the results, the numbers of resulting sessions between the combined interleaving approaches with a 1,440-minute inactivity timeout and the 30-minute inactivity variant are comparable, but naturally show negligible overlap – especially since one approach uses consecutive comparison between events and the other takes interleaving behaviour into account. Nonetheless, what is interesting about these approaches is their comparable number of sessions, which surely implies that they can reveal useful information about user behaviour and system usage.

The results overall are a challenge to interpret. The introduction of longer timeouts does not seem to have a particularly notable impact on the logical comparison method, whereas the shorter timeouts seem to impact relatively harshly, strictly interrupting, for example, the consecutive lexical variants. Selecting the correct timeout seems to be what is key here, which is dependent on what the desired outcome is. With a timeout of 75 or 180 days, the difference compared to the purely logical variants is negligible. The 1,440-minute inactivity timeout and the even lower variants seem to separate the sessions very strictly, creating far more sessions than the purely mechanical pendants. The 1,440-minute inactivity timeout sessions could be used to create either topically-related consecutive segments or related shorter segments within a confined time period. Both have their raison d'etre, but need to be considered in the context of the system. The same is true for the variants with a 14-day timeout – the results here are different compared to their non-combined counterparts and should cater to different system needs.

As discussed, the geometric sessions present a special case: their unique way of identifying session connection and session breaks leads to results that differ from the other combined approaches. For example, the **geomu14ac** variant identifies less sessions with a

14-day timeout than the **u2v10ti180ac** variant with its 180-day timeout – thus, clearly, the geometric approaches evaluate similarity more freely than the other logical comparison methods. Whilst this 'flexibility' could be seen as an advantage or a hindrance, it needs thorough analysis to understand why it is the case. It is not possible to dig deeper to gain a fuller understanding into the working mechanisms or why they differ from the other combined approaches within the scope of this dissertation, although needless to say the geometric sessions should be thoroughly evaluated before they are introduced to the productive environment. The supposition is that the geometric-session approach is unsuitable for this type of data or way of finding similarities between topics; using the calculated similarity between categories is less than ideal to achieve the degrading distance required by this approach, because even non-similar categories have some value of similarity to a reference category.

This concludes initial discussion of the various session approaches analysed in this dissertation. As presented, the session-identification algorithms deliver very different results. The difference within the approach types – mechanical, logical and combined – is dependent on different factors. Whereas for the mechanical sessions, apparently only the length of the session or the inactivity timeout influences the outcome, for the logical approaches there is greater complexity: there are the comparison method and the comparison context, even though there is no time constraint here. All factors influence the results for the combined approaches too, but evidently the timeout is the most crucial factor. Additionally, the comparison method has an impact as well: the differences between the lexical, semantical and geometric approaches are quite noticeable. Ultimately, the findings suggest that the various session types cater to distinct system needs – the different session definitions reveal disparate information about user behaviour, system usage and how users interact over time. These strengths and weaknesses are summarized again later in Table 7.1.

Chapter 6

Applying Sessions in Potential Production Scenarios

This chapter is the second part of the evaluation process in this dissertation. Where the previous section descriptively analysed the 135 session approaches to show the differences in the outcomes, this section actually utilizes the results in a selection of use cases. First, the dataset used in the case studies was prepared. A sample dataset was generated by true random sampling. It was ensured that the sample data was representative of the complete dataset. Only a selection of representative session approaches was used.

Having done this, three different use cases were implemented, all of these answering typical business questions and demonstrating how different session-identification algorithms will lead to different outputs. The first use case deals with the previously utilized category similarity. Here, the resulting session data is used to train sequence embeddings to determine relationships between categories. This experiment will show how the resulting similarities vary when using different session approaches. The second use case is a recommendation task. An example from the literature is reproduced with a variety of models. The third use case deals with user clustering. Using a DBSCAN [74] implementation, **user_ids** in the sample are clustered based on their session behaviour. The results are then discussed and analysed. Finally, the chapter ends with a summary of how the different session approaches impact the system evaluation and analysis.

Bucket	user_id share	interaction share
≤ 10	69.09%	21.77%
>10,≤ 30	21.66%	23.06%
>30,≤ 100	7.22%	23.01%
>100,≤ 500	1.86%	21.78%
>500	0.18%	10.39%

Table 6.1: Descriptive statistics for the interaction buckets in sampled dataset.

The sample is a 0.5% fraction of the original dataset's **user_id** population, taken via true random Bernoulli sampling. It is supposed to contain the same data for every **user_id**, therefore a sample of the **user_id** instead of actual rows was necessary. The resulting dataset contains 391,257 distinct **user_ids** with a total of 6,275,248 interactions. Descriptive values are shown in Table 6.1 to compare the sample statistics to the complete dataset.

The distribution of users and interactions is almost identical to the original dataset. The shares per interaction bucket are the same, indicating a strong overlap in properties.

This is also true for the other measures. Further statistical validation was not performed. At this scale, a correctly sized random sample should be valid enough to appropriately fit the data in its entirety. The assumption is that the sample is representative of the total number of identified sessions.

Approach	Sessions	∅Sessions	∅Interactions	∅Categories	B-R
u2vcti30cd	1,982,458	5.07	3.17	1.27	41.11%
u2vcti30cc	1,970,021	5.04	3.19	1.27	41.16%
u2vcti1cd	1,687,023	4.31	3.72	1.31	36.67%
u2vcti1cc	1,669,952	4.27	3.76	1.31	36.75%
ti30	1,616,825	4.13	3.88	1.46	34.43%
u2vcti1ad	1,522,246	3.89	4.12	1.34	32.81%
lti1cdb1	1,510,402	3.86	4.15	1.41	34.06%
u2vcti1ac	1,504,302	3.84	4.17	1.34	32.94%
u2vcti14cd	1,485,399	3.8	4.22	1.35	34.32%
geomu24cd	1,472,810	3.76	4.26	1.43	32.55%
u2vcti14cc	1,462,293	3.74	4.29	1.36	34.45%
u2vccd	1,454,600	3.72	4.31	1.36	34.12%
geomu24cc	1,448,111	3.7	4.33	1.43	32.63%
ti180	1,443,988	3.69	4.35	1.53	31.32%
u2vccc	1,429,489	3.65	4.39	1.37	34.26%
geomu24ad	1,393,275	3.56	4.5	1.45	30.65%
lti1adb1	1,375,339	3.52	4.56	1.45	30.92%
geomu24ac	1,371,057	3.5	4.58	1.45	30.76%
tfd	1,328,808	3.4	4.72	1.59	28.72%
u2vcti14ad	1,211,744	3.1	5.18	1.42	29.43%
u2vcti14ac	1,181,268	3.02	5.31	1.41	29.74%
lcdb1	1,140,546	2.92	5.5	1.61	28.94%
u2vcad	1,067,141	2.73	5.88	1.5	27.69%
u2vcac	1,024,081	2.62	6.13	1.5	28.17%
ladb1	726,069	1.86	8.64	2.03	20.62%
lti180adb1	722,392	1.85	8.69	2.03	19.64%

Table 6.2: Descriptive statistics for sessions in sampled dataset.

Table 6.2 lists the session approaches again with some of the most important descriptive statistics. The trends between the different approach mechanics resemble the trends in the original dataset. There are clear differences between the different variants, both structural as well as in terms of apparent content. The number of sessions is the clearest measure here, drastically decreasing (and following the trend in the original dataset) with an increasing timeout. The divergences between logical and mechanical sessions are clear as well. The identified sessions from the combined approaches vary strongly with the timeout used.

Logically, the variances for the other measures are quite high as well; on average sessions range from 5.07 to 1.85 while average interactions per session range from 3.17 to 8.69. The bounce rate behaves accordingly, decreasing in line with fewer identified sessions. This has a direct impact on the outcome of the case studies, because the preprocessing steps filter out all sessions with a single interaction. This is common practice, although it may drastically change the quantity of input data as can be seen in the table.

The number of average categories per session is relatively high for the longer-lasting combined sessions and the purely logical sessions, but comparatively they match the level of all the other approaches. It will be interesting to see how the contents of the sessions influence the algorithms. It is anticipated that the diversity of categories within the identified sessions will have a strong impact on the first two use cases.

6.1 Revisiting Category Similarity Vectors

This first task is all about identifying similar categories based on user category embeddings to validate the category tree and to show divergences in category similarity between session approaches. Earlier, this was used to generate the comparison mechanic for the logical sessions using category similarity based on category embeddings. The same procedure is employed now in a business case. The goal of this use case is to analyse the root category tree and to explore how it overlaps with the actual category similarity as perceived by the system users. The same assumptions are used for this exploratory analysis as previously: users are thought to visit categories based on their current information need, manifesting in the close proximity of similar categories as identified by the session approaches. While the previous calculation was performed on the complete user history in the hope that the algorithm would sort out any potential contradictions to that basic behavioural assumption with the sheer quantity of data, now the algorithm is trained on the individual sessions as identified by the session approaches.

The actual business case is this simple: By relying on the swarm intelligence of the users of the system, evaluating their sessions will find similar categories based on the session sequences. For every **category_id** of all 2,300 categories in the dataset, the top 25[1] similar categories are identified. Every category in this list has a root category; the number of different root categories in this top 25 list is calculated per **category_id** and then averaged per session approach. A high number of root categories indicates high diversity and potentially the need for changes to the category tree, while a low number indicates that the category tree works well according to the session approach now in use. Additionally, the calculated cosine similarity per top 25 categories per **category_id** per session approach is descriptively analysed using the minimum, maximum, average and the sum. The sum per all similar categories per **category_id** per session approach is also calculated. The cosine similarity is used as a way of showing structural divergences between the approaches – divergences in the calculated descriptive measures indicate different categories in close proximity in the sessions used for the category embeddings.

It is important to remind readers that this business case is completely dependent on the basic assumptions of a session as defined in Section 4.1. As the premise is that all identified sessions deal with one information need, it can, therefore, be assumed that the number of different topics in the form of different categories and root categories is limited. Should the user behave differently and work on more than one topic, then the session-identification approach in use will not have fulfilled its basic assumption. In practice, this will most likely be true for the majority of mechanically identified sessions; nonetheless, to be able

[1] The top 25 per categories are chosen somewhat arbitrarily. The average number of categories per root category is around 132. In order to have a comparable base, 25 categories per category was deemed a good enough number of categories for this comparison.

to implement this use case, it is assumed that the identified sessions deal with one topic[2]. With this in mind, the assumption of this use case – that different root categories are an indicator of a non-functional category tree – can be seen as a given precondition.Therefore, the business case can be understood more as a comparison between the session approaches and not so much one to provide a 'given truth' about the category tree.

Ultimately, the business goal is to have similar categories in the category tree, as this would be beneficial for the navigational properties of the system as well as for SEO. The defined goal here is to verify the category tree implemented in the page architecture. Having similar categories in the same branch is important for the design of the system because otherwise navigation might fail and search engines may have a hard time crawling the website. The premise of the business goal relies on the assumptions of the individual session approaches. For mechanical sessions, this is the time constraint: a user will deal with a certain topic within a fixed period of time or without interruption for a specific amount of time. For the logical sessions, this is self-explanatory since only similar categories are supposed to be visited. Actually, the outcome here will be very interesting; having only similar categories to calculate category similarity may lead to unexpected results.

In the main, this task is intended to show the divergences in category sequences between the different session approaches. The realization that these sequences will result in different category similarities is a clear indicator of the different output in this type of use case. The results may also show whether the assumptions of the session approaches hold true, or whether the mechanical sessions are apt to show very contradictory results.

Table 6.3 (on the next page) shows the results of the evaluation. It depicts the average number of root categories in the top 25 most similar categories per category per session approach as well as the number of tokens. Also presented are the multiple measures relating to the similarity score of the top 25 categories per category: the minimum, the maximum, the average and the sum. The last column contains the sum of all similar categories per category, equal to the size of the vocabulary per session approach. The tokens are the categories used in the embedding task.

The differences between the baseline (complete history) and the various session approaches as well as among the session approaches are quite big. At first glance, there seems to be a visible correlation between the number of tokens and the cosine similarity – the more tokens used as input for the algorithm, the lower the actual overall cosine similarity seems to be. This is somewhat reasonable since cosine similarity is in the range of [-1, 1]. The more tokens the algorithm has at its disposal, the more precise the development in the similarity relationships between the categories.

Interestingly, the minimum and maximum scores are rather low for the similarity score for the first 25 items when the number of tokens is high. This applies to the complete

[2] In a production environment, a use case like this can only be implemented if the identified sessions are validated. This means that their basic assumption should be qualitatively evaluated; this is not the scope of this dissertation. For the sake of being able to still apply the sessions in this use case, it is still assumed that all identified sessions deal with only one information need in order to show differences between the approaches.

Approach	Tokens	∅Roots	Min@25	Max@25	∅@25	Sum@25	Sum@Vocab
complete history	1,665,652	6	0.4	0.53	0.44	11.03	237.98
tfd	1,076,315	6	0.38	0.55	0.42	10.43	392.84
ti180	1,019,712	6	0.36	0.58	0.42	10.49	410.91
ti30	941,603	4	0.38	0.51	0.42	10.54	437.85
geomu24ac	793,745	2	0.58	0.7	0.62	15.46	618.76
geomu24cc	783,240	2	0.58	0.7	0.62	15.44	625.37
geomu24ad	770,745	2	0.61	0.71	0.65	16.24	668.57
geomu24cd	768,569	2	0.61	0.73	0.65	16.3	669.46
lti180adb1	1,171,349	2	0.71	0.81	0.75	18.79	741.16
ladb1	1,189,393	2	0.7	0.81	0.75	18.66	741.48
lcdb1	958,983	1	0.67	0.79	0.71	17.85	759.48
lti1adb1	759,979	1	0.67	0.8	0.7	17.44	773.25
lti1cdb1	958,983	2	0.66	0.8	0.69	17.21	784.25
u2vcac	726,461	3	0.72	0.82	0.76	18.88	799.2
u2vcti14ac	618,459	4	0.72	0.83	0.77	19.3	812.58
u2vcad	691,801	3	0.75	0.88	0.82	20.59	819.21
u2vcti1ac	525,641	4	0.7	0.84	0.78	19.51	825.18
u2vcti14ad	594,318	4	0.73	0.88	0.81	20.18	828.26
u2vccc	570,242	4	0.74	0.87	0.8	20.02	829.15
u2vcti14cc	543,720	3	0.74	0.86	0.8	20.04	832.45
u2vcti1ad	512,774	2	0.72	0.87	0.81	20.13	835.54
u2vcti1cc	490,092	5	0.75	0.87	0.8	19.96	836.69
u2vccd	555,248	3	0.71	0.86	0.8	19.98	838.81
u2vcti30cc	442,238	4	0.74	0.89	0.8	20.12	842.3
u2vcti14cd	529,982	4	0.75	0.88	0.81	20.36	843.43
u2vcti1cd	479,777	4	0.73	0.88	0.8	20.03	845.8
u2vcti30cd	434,272	3	0.74	0.87	0.8	20.04	856.78

Table 6.3: Overview of sequence embedding results per session approach. Depicted are the number of tokens and the number of average root categories per top 25 similar categories per category, averaged per session approach. Additionally, minimum, maximum, average and sum of the cosine similarity of the top 25 similar categories per category per session approach and the sum across all similar categories are shown.

history and the mechanical sessions. The similarity scores are also quite low and the total score overall is low as well, hinting at many negative values for the cosine similarity. Considering that these variants all have a high number of average root categories, there seems to be a connection between the diversity of categories in sessions and the ability of the algorithm to calculate similarity distances in this vector space properly. It is interesting to see that the mechanical variants are very close to the complete history as the baseline in terms of the presented values. Bearing in mind that the complete history delivered reasonable results (after manual evaluation) with more or less clear similarity scores, it can be assumed that with more data and adequate training time, the results for the mechanical sessions could be comparably reasonable as well.

In contrast to this, the lexical sessions (at least **ladb1** and **lti180adb1**) also have above 1m tokens but show scores very different to the mechanical sessions and the baseline's. The scores for the logical and combined approaches are higher overall, although the minimum and maximum are still close in most cases. However, while the overall sum is much higher, the sum for the first 25 categories is almost doubled for the highest scoring approach (**u2vcti30cd**) in comparison to the baseline. There also seems to be a rather clear distinction between the comparison contexts: the consecutive comparisons tend to score even higher than the interleaving approaches.

The lower the number of tokens, the higher the score seems to be. There does not appear to be any clear connection with the number of average root categories, although, again, an interesting observation is that the mechanical sessions / the baseline score the highest on average. In any case, the number of average root categories differs greatly between the approaches – these deliver clearly different results.

Overall, it can be safely assumed that the input data has a strong effect on the algorithm results. The results' implications are interesting: For one, it can be assumed that a high category diversity per sequence (i.e. per session) has a high impact on the algorithm output, leading to a more conservative similarity rating. This may be easily mitigated by simply feeding more data or more training iterations. Likewise, the more constrained sequences in terms of diversity seem to calculate similarity between the contained categories much more easily. This is especially visible when comparing the geometric approaches with the combined approaches using a consecutive comparison context and a shorter timeout. The divergence between these is very clear and attributable very likely to the fact that the geometric approaches contain more categories on average overall, while the combined approaches with the consecutive comparison context are both more self-contained and more easily interrupted because of the different comparison mechanic.

The self-containedness of these approaches in comparison with the interleaving approaches and the geometric variants is also a reason for the reduced number of tokens. As the preprocessing removes repeated consecutive categories and then removes sequences with only one remaining category, the probability of this happening is much higher for the consecutive comparison contexts (especially with a short inactivity timeout) because these approaches naturally tend to generate quite short sequences. For example, when the **u2vcti30cd** variant is compared to the **ti30** sessions, the difference in the number of tokens is quite large – where the **ti30** approach simply connects every consecutive category, similar or not, the **u2vcti30cd** approach only connects similar consecutive categories. If there is a repetition of categories (which is often the case), a sequence of exactly one interaction is more likely after preprocessing.

From an overall perspective, this is a clear indicator of user behaviour regarding the session approaches. In contrast to the basic assumption for the mechanical sessions – that users work on one information need before working on another, separated by breaks – the users in this dataset seem to work on multiple unrelated topics. This behaviour leads to more self-contained sequences for the combined approaches and more diverse sequences for the mechanical sessions, in the end resulting in different embeddings. Another indicator of this is seen in the **lcdb1** lexical variant: connecting consecutive interactions only when the root category is the same, the results are still different to the mechanical sessions even though the number of tokens is comparable.

Of course, these assumptions are only based on the algorithm output with several caveats. For one, the data input was rather low for an embedding algorithm. While more data does not always improve the output, particularly as more data always means more noise [299], it can still be supposed that the results would be smoother and better for comparison purposes with more data or even longer training. Then again, the preprocessing is an essential step when applying such an algorithm [197]. Seeing as the impact of the preprocessing on the different session approaches is quite big, this is a strong argument to choose a fitting approach carefully when using machine learning. Another caveat relates to the way the logical approaches use similarity based on the same algorithm utilizing the baseline data, and how this could represent a self-fulfilling prophecy as regards user behaviour, this is a point that should be kept in mind when interpreting the results.

In summary, using the same input data plus applying the same data preprocessing seems to have a strong impact on the output of the algorithm. Logically, the results per session approach are not completely different in terms of which categories are similar to which category, but the differences are still very noticeable. For example, across all session approaches, the distinct count of all similar categories for the top 25 similar categories for the category *Smartphones* is 93, indicating at least some diversity but also some overlap between the various approaches.

Taking a bird's-eye perspective on the shared content related to the similar categories across all categories, unsurprisingly, the baseline shares the biggest overlap with the mechanical categories. When comparing the top 25 similar categories across all categories, from a possible complete overlap of 58,000, around 27,000 are shared with the mechanical approaches. The lowest overlap is around 21,000 shared similar categories with some the combined approaches. This is also the lower overall boundary: around 20,000 to 21,000 categories are shared, no matter the session approach. The higher overall boundary is different though: the logical semantical and combined approaches share up to around 50,000 overlap. Interestingly, this seems to be dependent on the comparison mechanic – the variants using lexical, semantical or a geometric comparison all only share around 28,000 categories. This is a strong indicator that the comparison mechanic (i.e. using either vector embeddings or geometric distance) creates structurally different sessions.

This gets even more interesting when comparing the mechanical sessions with the other approaches. They seem to have a high overlap of similar categories among each other (around 40,000) but not so much with the semantical combined approaches (around 22,000). Nevertheless, the geometric approaches seem to be more similar: here, the overlap is higher with around 30,000 categories among the top 25 per category. These findings confirm the suppositions made previously.

Figure 6.1 takes an example of shared categories in the category *Smartphones*. The figure shows the connection between the session approaches. If two approaches share a similar category in the top five similar categories, they are connected. A connection is only made when they share at least two entries to make the stronger connections more visible. As can be seen, there are some interesting connections that indicate a certain proximity between approaches. For one, there seems to be a great overlap between the logical lexical variants – they share a high number of similar categories among them, but not so many compared to the other approaches. The baseline only shares more than one category with this approach group, but not with any other approach. This is an interesting indicator of their structural similarity. Likewise, all the logical approaches obviously share a high level of overlap; this makes sense considering that they all use the same mechanic, although the sessions may vary greatly in length. The mechanical approaches also have a high overlap among each other, but are also somewhat connected to the geometric variants. Overall, the figure underlines the findings presented for this specific example. In the theory, the

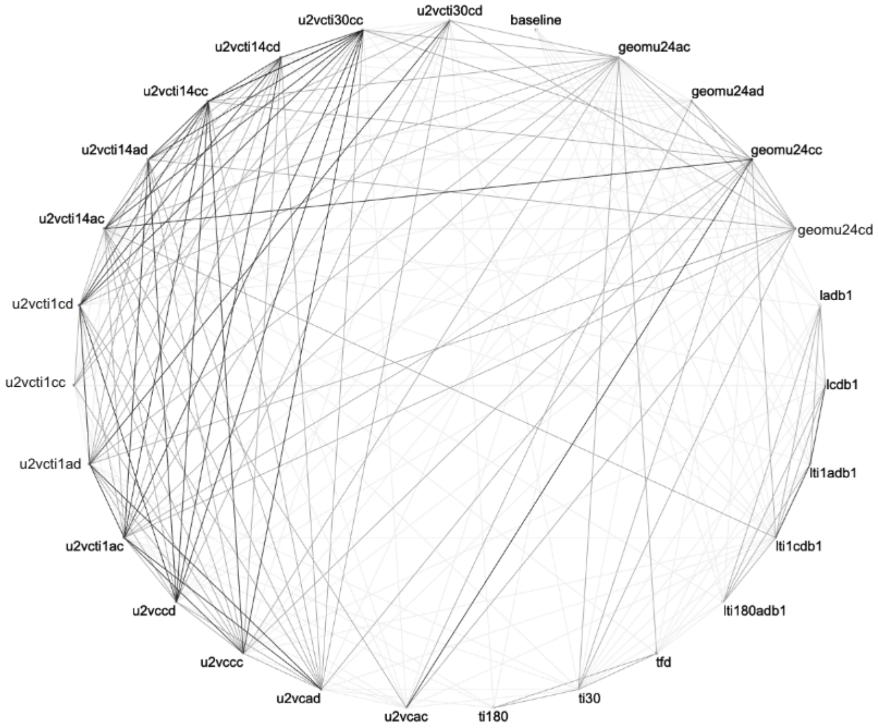


Figure 6.1: Chord chart of similarity connections between the session approaches for the top five similar categories of the category *Smartphones*.

figures look similar for all categories, but in practice the findings are largely unreadable due to all the connections. Therefore, the example of *Smartphones* has been chosen to illustrate how it would look in practice.

This concludes the discussion on the category sequence embeddings use case. The results underline the discoveries from Chapter 5. They clearly show differences in the algorithm output when using structurally different input data, although the actual data is still the same. As other research has reported, many factors affect the building of word or item embeddings [72, 135]. The results in this dissertation clearly underline these findings.

6.2 Predicting What's Next: Session-Based Recommendation

The second use case tested in this dissertation is an example from the literature. As was mentioned in Section 2.5, recent years have seen quite a rise from sequential models like recurrent neural networks (RNN) and their subvariants such as long short-term memory networks (LSTM). Delivering impressive results and an enormous predictive power based on sequences of data, these models continue to attract a lot of attention. Although review articles state that simple baselines may deliver comparable or even better results [156], the actuality and potential of these models is undisputed. This dissertation reproduces the model proposed by Ruocco et al. [225] along with some of their proposed baseline models (among other things, a traditional session-based RNN [102]).

Technically, the use case is not a prediction task but one modelled as a session-based recommendation. The models are trained on the sessions of users and evaluated on their last session. For every interaction of this last session, the model recommends a set of items – categories in this implementation – and ranks them accordingly. Theoretically, only the correct **category_id** of the next interaction is the target, so the results are evaluated as a binary classification task using recall and mean reciprocal rank per interaction step in the session.

What is special here is the fact that the tested models all stay the same while the input data is changed. The different session approaches identify different sessions, which in turn are used for training and testing the model. By taking into consideration the assumptions that guide all the approaches, an estimation can be made of how the input data affects the way the model predicts what the next item is (or multiple item recommendations) in a respective session for that specific session approach. Ultimately, this makes it possible to state how well the model is able to generalize on the basis of the input data.

Four different models are tested: most popular (mp); k-nearest Neighbours (knn); a traditional session-based RNN (pRNN) [102] and the model proposed by Ruocco et al. [225], using a session-based RNN alongside information taken from embedded historical sessions (iRNN). All of these were also used in the original article, whose implementation this dissertation reproduces. Multiple models are tested as a sanity check and to see how the different session approaches perform under different conditions. While both deep learning models are broadly similar, the two baseline models work differently. Technically, this type of variety is not required to show the differences in the input defined by the session approaches, but overall it is beneficial because it widens the perspective.

The mp is a baseline model that simply recommends the most popular items in the dataset. Here, this means that all **category_ids** are sorted by occurrence and the top-k items are recommended at every interaction step in a session. Furthermore, another baseline model, knn, determines the number of co-occurrences of items per session. The recommendation is then a list of **category_ids** with the highest co-occurrences per **category_id** of the respective interaction step. The session-based pRNN is a sequential model: it bases its recommendations on which **category_ids** are visited in a session, how, and in which order. On conclusion of the session, this user information is discarded – in a new session, the model makes a prediction based solely on the new session, using no contextual information at all. The same technique is used for the iRNN, although Ruocco et al. came up with the idea to improve the session-based RNN by creating embeddings for previous sessions and including them as starting input for the session-based recommendations.

Since there seems to be no standard evaluation of session-based recommendations [156], this dissertation follows the evaluation of the original work, splitting user's sessions 80:20 – 80% of identified sessions used for training and 20% used for testing. In practice, for all runs, this means that not only is the structure of the input data in the training set different, but that the data for the test set is also. For example, session approaches that are able to identify interleaving behaviour would also generate recommendations for that interleaving behaviour – the model is tested on the identified session events, which do not necessarily have to be in chronological order. This comes with some limitations: in a productive environment – as the actual session context becomes usable only after the session-identification process – the session identification would always run before the recommendations are generated by the model.

The limitation this evaluation set-up poses is interesting as is the overall question of how to manage it. In terms of the 80:20 split based on user sessions, this could result in a big difference in the training- and test-sets per user per session approach. In practice, it would probably be more reasonable to run the test on a chronologically ordered set of interactions – without the boundary of a session (i.e. training everything up to a certain point in time and testing everything again afterwards). Then again, the training set is structured according to the session approach. The session approach defines the sequence of interactions, so it is actually reasonable to generate predictions for such a specifically defined session. The session approach delivers the context for which the model generates output. Testing the algorithm on non-structured data would actively hinder the delivery of high-quality results: one can see the session approach as part of the algorithm in this case, amplifying the quality of the output (or at the very least changing it). From this perspective, using the 80:20 split as suggested seems reasonable. Therefore, this dissertation stays with the original 80:20 split and the respective evaluation, but mindfully respects its limitations while interpreting the resulting evaluation measures.

With the four models and the design for the experiment ready, some pre-emptive assumptions can be made. From the findings of the previous chapter and with knowledge of the mechanics of the respective session approaches and what the identified sessions may look like, some potential outcomes seem more probable than others. This is especially true for the baseline algorithms. For instance, it is far more likely that a good hit rate will be achieved by using the most popular items across all sessions for the mechanical sessions, than for the logical or combined ones. This makes sense: the category mix is more or less random for the mechanical approaches, meaning that a popular category might be visited in any session and at any point in time. Using logical mechanics, this is not the case: here, the session may revolve around a specific topic, only including related categories that do not necessarily have to be among the most popular ones, resulting in a very low recall. Contrary to this, a logical session dealing with a popular topic might achieve very good results; still, the mechanical variants are anticipated to perform better results here. This is also true for the geometric approaches since their comparison mechanic is not as strict as the logical ones, potentially leading to more mixed categories in a session.

The knn baseline is more difficult to assess. Potentially, logical sessions may have an advantage here. The coupling of categories that often appear next to each other for use in generating recommendations are likely to work better when all the categories are related anyway, which would be the case for logical sessions. On the other hand, the base assumption of the mechanical sessions – that the inactivity timeout represents a topic break – may lead to similar performance. It is difficult to predict the results of the RNNs beforehand. Under normal circumstances, it could be assumed that the session approaches using a logical comparison mechanic would have an advantage here: using topical closeness to create recommendations seems to be more promising than simply using chronological sequences of categories that are only assumed to be similar. In practice this means that the RNNs are trained on the same categories they generate as predictions, whereas the mechanical approaches are trained on every data point. It will be interesting to see in this context whether the underlying assumptions about the session approaches actually hold true.

Approach	Users	Sessions (Training)	Sessions (Test)
ti30	30,557	217,659	68,011
ti180	30,005	203,258	64,126
tfd	30,300	194,378	62,045
u2vccc	25,615	159,969	51,318
u2vccd	26,806	169,757	54,310
u2vcac	23,647	98,327	34,885
u2vcad	25,742	121,119	41,584
ladb1	22,818	64,665	25,762
lcdb1	27,478	149,452	49,465
u2vcti30cc	27,647	203,620	63,240
u2vcti30cd	28,322	207,667	64,546
u2vcti1cc	26,777	182,606	57,547
u2vcti1cd	27,590	188,919	59,475
u2vcti1ac	25,437	166,119	52,863
u2vcti1ad	26,380	174,006	55,211
u2vcti14cc	26,221	164,345	52,617
u2vcti14cd	27,284	173,461	55,405
u2vcti14ac	24,008	119,523	40,416
u2vcti14ad	25,449	135,322	45,003
lti1cdb1	28,156	183,490	58,339
lti1adb1	26,867	166,874	53,611
lti180adb1	22,790	64,917	25,845
geomu24cc	27,243	172,947	55,320
geomu24cd	28,030	182,099	57,950
geomu24ac	26,197	160,991	51,848
geomu24ad	26,835	169,568	54,289

Table 6.4: Overview of number of sessions and users in the dataset for the recommendation task.

Table 6.4 shows the number of sessions for all approaches, their respective training- and test split, as well as the number of distinct users. It is clear that the preprocessing (i.e. cut-off after 20 interactions, no repeated consecutive categories) has a strong impact on the overall number of sessions. The notable differences between the session variants (indicating potential structural differences) seems reasonable considering the different mechanics and the fluctuating bounce rates as well as the different comparison contexts. The number of users stays on a similar level with minimal divergences across the approaches. It is interesting to see that more users are retained by the mechanical variants after preprocessing while the logical variants seem to lose more data in general.

The effect of fewer sessions is particularly apparent for the lexical variants allowing interleaving behaviour (combined approach as well as purely logical). These approaches show a very limited number of sessions for both the training set and test set, but this again is reasonable since they have the lowest overall number of sessions of all the approaches with the potential to result in long sequences of repeated categories for many users. Yet, since the number of users remains comparable with the other approaches, it is probable that the respective lexical variants simply connect the same categories in sequences, resulting in less sessions per user in both datasets. This is an intriguing finding for both the assumptions of the models and the session approaches: How is the iRNN going to perform if the chronological ordering of the sessions is disturbed by the preprocessing? and How is is the dependency on previous sessions going to develop when the ordering is unrelated to topic recency as it is for the logical sessions? Whereas the ordering of the mechanical sessions is straightforward because of their basic assumption, the logical sessions assume a different ordering of events, at least the interleaving variants do. This is equally complicated for consecutive variants: usually, consecutive sessions identified by a logical mechanic are assumed to deal with different topics when there is no combination mechanic involved.

At this stage, it is not really possible to estimate beforehand what impact the number of variable sessions will have on the model output. In all likelihood, the model would be able to deliver better results with more training data. Nonetheless, since preprocessing is essential in the preparation process, it was decided to keep the size of data as is; as the session approach itself is responsible for the structure of the input data, here, therefore, the number of sessions can already be seen as an example of the approach's impact on the model output. It will be interesting to see if there is a correlation between the quantity of data and the model output.

Table 6.5 lists the results for all session approaches and all models. The reported values are averaged across all session interaction steps up until the maximum length of 20 interactions. Highest overall values are highlighted. All models were trained until the evaluation measures no longer improved up until a maximum of 10 epochs. The majority of approaches reached this point around epoch seven, after which the results started to diminish again.

	mp				knn				pRNN				iRNN			
	MRR @1	MRR @5	R@5	R@20	MRR @1	MRR @5	R@5	R@20	MRR @1	MRR @5	R@5	R@20	MRR @1	MRR @5	R@5	R@20
ti30	.3527	.3761	.4167	.4907	.3487	.4032	.5157	.656	.4527	.5351	.6575	.7662	.4734	.5589	.6844	.7831
ti180	.3286	.3529	.3943	.4693	.3283	.3803	.4886	.6309	.4344	.5156	.6365	.7474	.4549	.5394	.6636	.7657
tfd	.3109	.3359	.3783	.4566	.3106	.3613	.4673	.6112	.4223	.5035	.6229	.7309	.4404	.5238	.6472	.7515
lcdb1	.3724	.3518	.3987	.4748	.3204	.3795	.4992	.666	.4485	.5275	.6503	.7835	.4622	.5403	.662	.791
ladb1	.2174	.2417	.2811	.3608	.2127	.2674	.3788	.5581	.3045	.3748	.4923	.6633	.3109	.3769	.4869	.6556
u2vccc	.4341	.4589	.5009	.5726	.4257	.5034	.6547	.8149	.5943	.6855	.8238	.9306	.6057	.6965	.8335	.9342
u2vccd	.4345	.4592	.5011	.5732	.4281	.5085	.6669	.8289	.5913	.6864	.8315	.9412	.6076	.7	.8409	.943
u2vcac	.3516	.3784	.4219	.4998	.3435	.4277	.5935	.7713	.5238	.624	.7793	.9146	.5361	.6313	.7799	.9139
u2vcad	.3506	.3767	.4195	.5014	.3474	.4368	.6145	.8027	.5275	.6364	.8068	.946	.5422	.645	.8067	.9436
geomu24cc	.3832	.4072	.4489	.5235	.3759	.4379	.5641	.7231	.5151	.5997	.7263	.8404	.5277	.6135	.7412	.8494
geomu24cd	.3829	.4072	.4496	.5258	.3807	.4427	.5698	.7327	.5101	.595	.7223	.843	.526	.6128	.743	.8554
geomu24ac	.3783	.4016	.4416	.5164	.3715	.4338	.5605	.7181	.5135	.5982	.725	.8402	.5283	.6134	.741	.8493
geomu24ad	.3786	.4024	.4435	.5198	.3732	.4371	.5673	.7291	.5116	.5976	.7262	.8435	.5251	.6115	.7401	.8516
lti1cdb1	.3956	.4222	.4689	.5418	.3885	.452	.578	.7375	.5204	.6034	.729	.8478	.5367	.6205	.7472	.8598
lti1adb1	.3802	.4057	.4495	.5226	.3738	.4357	.5598	.7209	.5088	.5927	.7188	.8373	.5218	.6058	.7316	.8454
lti180adb1	.2232	.2475	.2871	.3655	.2187	.2726	.3821	.5587	.2414	.2837	.3586	.4952	.3168	.3826	.4917	.6565
u2vcti30cc	.5071	.5288	.5664	.6308	.5067	.5788	.7199	.8591	.6195	.7092	.8431	.934	.6384	.7263	.8568	.941
u2vcti30cd	.5095	.5311	.5684	.6325	.5099	.5842	.7287	.8672	.6201	.7094	.8427	.9353	**.6391**	**.7287**	**.8623**	**.9453**
u2vcti1cc	.4695	.4931	.5332	.6032	.4625	.5388	.6872	.8354	.6117	.7005	.8335	.9311	.6257	.7147	.8482	.9374
u2vcti1cd	.4715	.4949	.5349	.6044	.4664	.5446	.6975	.8457	.6083	.7001	.839	.9381	.6285	.7185	.8531	.9431
u2vcti1ac	.4535	.4766	.5159	.5869	.4461	.5223	.6735	.8268	.6043	.6947	.83	.9311	.6184	.7074	.8401	.9345
u2vcti1ad	.4526	.4755	.5146	.5855	.4466	.526	.6825	.8389	.5982	.6928	.8359	.9408	.615	.7083	.8481	.9438
u2vcti14cc	.439	.4637	.5057	.5776	.4322	.5083	.6581	.8149	.5975	.6874	.8234	.9274	.6084	.6977	.8322	.9303
u2vcti14cd	.4399	.4644	.5063	.5784	.4339	.5129	.6702	.8284	.5928	.6857	.8275	.9365	.6092	.7007	.8395	.9393
u2vcti14ac	.4044	.4295	.4715	.5426	.3962	.474	.6272	.7883	.5805	.6711	.809	.9228	.5877	.6767	.8127	.9211
u2vcti14ad	.3982	.4229	.4647	.5375	.3924	.4742	.6372	.8092	.571	.6693	.8201	.9412	.582	.6781	.8259	.94

Table 6.5: Overview of model results for recommendation task. Highest overall results are highlighted. Abbreviations: R = Recall. MR = Mean Reciprocal Rank.

The reported results are quite interesting. From an algorithmic point of view, the statements made in the original work by Ruocco et al. [225] could be confirmed. No matter the session approach representing the input data, the proposed inter-intra RNN delivers the best results. From a logical point of view this is reasonable, since information from earlier session contexts as well as the current context should make it easier to make predictions (or recommendations) for the current behaviour. A particularly compelling insight is that almost all session approaches are able to generate impressive results using this model. Although previous sessions would not necessarily have been dealing with the same topic (i.e. for the mechanical variants), the results are still very good.

The reported values for R@20 are quite impressive. It seems that in the majority of sessions, the target category is among the list of the recommended top 20 items. Even R@5 does not seem to be too far off – the difference is modest against R@20. The MRR is even more promising. In over half of the test cases, the iRNN model seems to be able to predict or recommend the correct target item for logical approaches. The mechanical variants, including the lexical logical baselines, achieve slightly less impressive results. There is a clear difference in performance even for the iRNN. The numbers for MRR@1 for the lexical variants are quite surprising. Here, the low performance was not expected, but seems somewhat reasonable; putting all **category_ids** of the same root category into one session without regard to actual similarity may introduce an error margin bigger than originally anticipated. It seems that actual topical connections are interrupted here. That the inactivity timeout sessions appear to perform better is very interesting, indicating a more compatible mixture of categories and actual temporal proximity of logically connected topics. This is confirmed by the combined approach using the lexical matching actually performing on par with the other logical approaches: **lti1adb1** and **lti1cdb1** indicate that using one day as a timeout helps put topically related categories together, no matter how liberal the mechanic is. This is doubly underlined by the very poor results of **lti180adb1**, again due to its connecting potentially unrelated categories over a long period of time.

Nonetheless, the difference between mechanical sessions on the one hand, and the logical and combined approaches on the other, is relatively big and ought to be acknowledged. There is a clear improvement in predictive power when using sessions constructed with logical mechanics. Another important observation is that there seems to be a divergence between a) the comparison contexts and b) the logical and combined approaches. For one, the direct comparison always seems to perform slightly better than a comparison based on all categories in the respective session. This could indicate that recent items attract higher relevance, in that users work on information needs, formulate additional goals, and then move on to different tasks. This is reflected in these sessions: categories may end up in different logical sessions because the most recent item determines the current similarity map for any comparison with a future interaction. Moreover, the model seems to work better with the consecutive approaches than the approaches allowing interleaving behaviour, although it seems that the direct comparison has an even greater impact here than the actual context. This could also be why the combined approaches appear to achieve better overall results than the purely logical approaches. Although the difference is rather slight here, the recency factor introduced by the timeouts seems to be beneficial. It is also highly interesting that the shorter timeouts seem to work better than the longer timeouts. This is another indicator of the dominance of recent categories in determining current behaviour.

These observations are not only true for the iRNN, but seem to be similar for the other models. The pRNN follows a very similar pattern, only usually with slightly less favourable results. The difference is again very reasonable, considering that the iRNN simply has more information available than the pRNN. This as well could be interpreted as another argument for this recency dominance theory: context knowledge of the previous session can lead to improvements due to the information not being removed after the session is over. In theory, this should be more apparent in the sessions with a shorter timeout. For example, it is assumed that the 14-day timeout gains less advantageous knowledge of the previous session than a 30-minute timeout. The effect is visible in the table (i.e. when comparing the measures between the models), but the difference is quite small and could be due to other factors as well. Some of the approaches even generate better results for R@20 with the pRNN.

The previously made assumptions about how the baselines would perform seem to not hold true for this particular testing scenario. The most popular categories in particular do not seem to work for the majority of approaches. It is intriguing to see that the **ti30** sessions (and the mechanical sessions in general) perform very badly here; in comparison to the combined approaches with a 30-minute timeout, this seems very surprising. Since the assumption is that many of these are simply the same sequences, one would usually expect to see them corresponding at some level. Then again, the anomaly could be put down to preprocessing and the removal of repeated consecutive items, whereby the similar sessions between them have been removed and only the differing ones remain. In these sessions, the combined approaches may perform training better than the mechanical variants.

The same is true for knn, although the overall results seem slightly better than the most popular recommendations. The mp model has a better average MRR@1 in most cases though, indicating a higher hit rate by simply using the most popular **category_id**. This is a good example of how the overall system affects the performance of the model – in around one-third of all cases, simply predicting the **category_id** with the overall highest interactions is sufficient. Due to this, the performance of the models should be taken with a grain of salt; if the majority of interactions is on the same **category_id** anyway, recommending the correct target item should be quite easy. The trends otherwise match all the other models. For R@20, the results are relatively close to the performance of pRNN. The same is true for R@5, although the difference is slightly more obvious. Comparatively, the MRR does not perform quite so well. Overall, the general trends for the measure are very similar across the models. The performance increase from baselines to better adjusted models is relatively stable – in general, the models seem to work well with every approach and the performance increase is comparable across all approaches.

Some approaches are curious though: as an example, the **lti180adb1** lexical logical session approach allows interleaving behaviour, and to all intents and purposes performs comparably to the baseline lexical logical approach **ladb1**. The difference comes in the form of the 180-day inactivity timeout and the associated handling of interactions on pages with no meaningful **category_id**. While the assumption is that the inactivity timeout has almost no effect here (the 180 days with interleaving behaviour is probably only breaking sessions in very specific edge cases), the difference in handling of any interaction with a non-meaningful **category_id** results in sequences interrupted by these **category_ids**. Meanwhile, the baseline with no timeout has less of these interruptions, resulting in sequences with meaningful **category_ids** only. This difference seems to cause massive problems for the algorithm. The conjecture is that the non-meaningful items interrupt the input sequences and, therefore, dilute the actual meaningful sequences of related **category_ids**. There is an intriguing divergence here between knn and pRNN for recall: comparatively, the iRNN delivers the better results for recall, but here the knn seems to outperform the pRNN. This is highly interesting, especially since the pRNN MRR results are better. It would seem that the pRNN is more often able to predict the target correctly or at least among the top five, while the knn model predicts the target item more often among the top 20 in general. This is the only case where this happens though.

Returning to the number of sessions, there is a clear pattern here. The number of sessions used for training the models indeed correlates with the model outputs, although there are some limitations to this statement. On the one hand, the trend is clear because, obviously, the more training data used, the better the results will be. This seems to hold true independent of the model, although some models may still be able to deliver solid results even with less sessions. Likewise, a high number of sessions does not guarantee good performance, as can be seen with the mechanical sessions. Nonetheless, the low number of sessions for **lti180adb1** and **ladb1** seem to somewhat explain the comparably poor results. Interestingly, the same cannot be said for **u2vcac**, which also had considerably fewer sessions than the rest. Here, the algorithms still perform well enough to deliver results comparable to other approaches, especially when compared to the other comparison contexts with the same comparison mechanic. In conclusion, it can be surmised that a combination of different effects, as discussed, is responsible and that probably that the structure of the sessions is more important than the quantity of training data alone. The models seem to be very sensitive to the session structure.

Another curious case is the performance of the **u2vcti30cd** and the results for the combined approaches with a shorter timeout in general. Compared to all other approaches, **u2vcti30cd** achieves the highest values for all measures and all models. Even for the two baseline models, the reported results are impressive. On a comparable level is its counterpart **u2vcti30cc** with the complete session history comparison mechanic, but as already reported the direct comparison seems to work better in most cases.

To conclude the applied recommendation task, this section has shown that the combined approaches using the embedding solution to compare similarity of interactions work best no matter the algorithm. The mechanical variants and also the approaches using the lexical matching of root categories fail to deliver results of the same quality. The geometric approaches and also the purely logical variants using the embedding solution are better, but still not quite as good as the combined approaches. Several factors are potentially responsible for this outcome. These will be discussed in more detail in Section 6.4.

6.3 Clustering Users Based on their Session Behaviour

The third use case is a simple clustering task to show how different input data may lead to varying interpretations of the user population and, to some degree, of user behaviour. The goal is to identify certain behavioural clusters depending on how users in the data interact in sessions with the system. Knowledge about such clusters and their potential for different behaviours enables system owners to target users in very specific ways. For example, users with low recency may be targeted more often, while users with higher potential in this respect may be contacted far more specifically; if they return to the system anyway, they may be guided more towards their interests rather than simply being tempted back to the system.

In the end, the clustering task here is very simple and straightforward. Having a set of session-related features per **user_id** should be sufficiently distinct to show the differences between the different session approaches. Based on the results of the previous sections there are multiple assumptions about how the resulting clusters may look. For one, the number of clusters will probably be quite low for all session approaches simply because the majority of users do not behave that differently; with a low number of overall interactions and very likely only a few sessions, users will probably end up in the same cluster regardless of the session approach. All session approaches may show an overall similarity, the assumption being that the majority of users with a low number of interactions in total will end up in the same cluster regardless of the session approach.

Aside from this, the supposition is that the different session approaches will generate various clusters that may share some similarities. It will be interesting to see which session approach users with a high number of interactions end up in: having a high number of sessions in the **ti30** approach because of many regular visits to the system across the year does not necessarily equate to a high number of logical sessions. The features of this sample user differ greatly.

As the aim of this dissertation is to highlight the distinctive features per session approach, each feature directly relates to session behaviour. The dataset used for the clustering always contains the same **user_ids**, but the actual features are based directly on the resulting sessions of the respective session approach. First and foremost is the number of sessions per user, this being possibly the most important measure to indicate any potential cluster. The number of sessions almost always has some effect on the other measures as well. Likewise, the average number of interactions per session is equally meaningful. Depending on the mechanics of the session approach, the identified sessions are expected to contain either a lot or only a few interactions.

Table 6.6 shows the number of clusters, the silhouette score and the respective number of data points suspected to be noise per session approach. As a reminder, the silhouette score is an indicator of the quality of the resulting clusters, that is, measuring to what extent the clusters are well-distinguished and how much they overlap. It is apparent immediately from the silhouette score that the clustering for all session approaches seems to identify a somewhat decent cluster. A score above 0.5 indicates at least reasonably dense clusters. While the values vary only slightly, some observations can still be made here, particularly in relation to the number of data points identified as noise by the algorithm. Likewise, the number of clusters does not to change drastically, but there are some differences.

Approach	Clusters	Silhouette Score	Noise
ti30	73	0.5947	22,546
ti180	73	0.5875	19,658
tfd	72	0.586	19,011
u2vccc	78	0.5845	24,804
u2vccd	74	0.5428	20,098
u2vcac	75	0.661	9,712
u2vcad	74	0.6656	15,670
ladb1	67	0.7275	9,483
lcdb1	74	0.6603	21,230
u2vcti30cc	76	0.5388	31,953
u2vcti30cd	76	0.5508	37,227
u2vcti1cc	80	0.5253	27,998
u2vcti1cd	74	0.516	27,264
u2vcti1ac	73	0.5488	23,871
u2vcti1ad	76	0.5942	22,440
u2vcti14cc	73	0.5498	21,064
u2vcti14cd	74	0.5462	20,593
u2vcti14ac	79	0.6492	16,915
u2vcti14ad	74	0.6343	16,773
lti1cdb1	82	0.6341	22,164
lti1adb1	74	0.5993	19,103
lti180adb1	68	0.7333	4,371
geomu24cc	78	0.6056	21,158
geomu24cd	74	0.5956	20,852
geomu24ac	76	0.5987	24,564
geomu24ad	74	0.6154	19,047

Table 6.6: Overview of clusters per session approach. The table shows the amount of overall clusters, the silhouette score as well as the amount of data points that could not be associated with any cluster.

The correlation between the silhouette score and the number of noise data points seems obvious. The approaches **ladb1** and **lti180adb1** have the highest score and the least amount of identified noise. Again, this does not necessarily constitute higher-quality clustering, as this very much depends on the input data and the preprocessing in terms of normalization. Under the current circumstances, these approaches seem to provide the clearest results though. At the other end of the scale, **u2vcti30cd** has the highest number of noise data points, but not the lowest silhouette score – the correlation seems to be not that straightforward.

Aside from the silhouette score and the number of noisy data points, the number of clusters is relatively high. This is reasonable considering the used hyperparameters: a minimum of 1,000 samples per cluster, with a neighbourhood of at least 100 samples in order to be considered a cluster centroid, is likely to end up in a higher number of clusters. In a separate run with higher values for these parameters, the number of clusters was far lower per session approach but more or less in the same relation as shown in the table. Interestingly, the silhouette score decreased drastically and the number of noisy data points increased, which is why the current setting was used to show the differences per session approach. This is an interesting observation though. The hyperparameters required for an HDBSCAN obviously have quite a big impact on the clustering results – even greater than the actual session approach. The same is true for any normalization applied beforehand. While the effect of these steps is quite apparent, the impact on the session approach appears to be less so in comparison, although still always discernible. Depending on which normalization is selected, the impact of the session approaches either becomes clearer or not immediately apparent. While the actual hyperparameters seem to conserve the impact of the session approaches as regards the relationship between them, the quality appears to decrease.

The number of identified clusters seems quite large, and considering that the algorithm only uses two features, it is surprising to see so many apparent subgroups of reasonably distinct behaviour. What is even more surprising is that the number of clusters only differs slightly between what are overall very different session-identification approaches. It could be that the relationship between the number of sessions and the average interactions per identified session leads to a different number of clusters based on the session approach, but from the number of clusters alone, this difference is not discernible. Looking at the structure of the clusters, the observations are also very similar. The biggest cluster per session approach always contains around 30,000 **user_ids**.

Table A15 displays the 10 biggest clusters per session approach as well as the number of data points assumed to be noise. The table shows the share of **user_ids** assigned to the cluster and the average number of input features used: that is, the average number of sessions, with the average number of interactions per session per cluster, per session approach. The results show a clear pattern with some surprising outliers: almost all session approaches share the same clusters in their top 10, but in varying order and with different shares of **user_ids**. By using only the two features – sessions per user and average interactions per session per user – the results clearly show how different yet similar the session approaches are overall. The biggest cluster almost always contains the users with only one session and exactly three interactions. Likewise, the other clusters among the top five are also almost always made up of users making between two and five interactions overall. The number of sessions varies slightly, but the cluster statistics are identical.

This simple clustering of the two features has some interesting implications. It is basically a segmentation into interaction and session buckets: a perfect clustering would have the same cardinality as the number of distinct combinations of sessions and average interactions. This is logical. For the algorithm results, this means that the clusters perfectly represent the differences of the session approaches, because the algorithm technically detects the same clusters per approach. For example, the cluster (or bucket) with one session and three interactions seems to be the biggest cluster for all approaches except **u2vcti30cc** and **u2vcti30cd**; for these two approaches, the bucket with two sessions of only one respective interaction is bigger. It can be assumed that there is some overlap here which could be attributed to the session approach.

The associated noise per cluster is also quite revealing. Taking the view that these are users that could not be put into any of the other clusters or interaction/session partitions, these are highly likely to be the combinations of features with a cardinality below 1,000. Since the statistics of these noisy users differ greatly across the session approaches, it is probable that there are differences between them regarding the distribution of sessions and the respective interactions per session per user. Under the set hyperparameters, the clustering as is makes sense. With wider parameters (i.e. a minimum cluster size of 10,000 and a more generous minimum number of close neighbours), the session/interaction combinations would be merged earlier, introducing bigger clusters but with more ambiguity.

Completely allowing the algorithm to determine the cluster size (by using the default values[3] leads then logically to a very high number of clusters.

Approach	Clusters	Silhouette Score	Noise
ti30	19,611	0.3924	72,232
ti180	20,040	0.3871	75,508
tfd	19,968	0.3884	77,326
u2vccc	20,606	0.4114	78,627
u2vccd	20,711	0.4216	77,276
u2vcac	20,545	0.3953	81,545
u2vcad	20,638	0.3904	82,309
ladb1	20,932	0.3687	82,350
lcdb1	20,767	0.3643	81,242
u2vcti30cc	19,295	0.4391	66,225
u2vcti30cd	19,355	0.4411	65,642
u2vcti1cc	20,228	0.4477	73,884
u2vcti1cd	19,991	0.4434	74,538
u2vcti1ac	20,172	0.4399	73,235
u2vcti1ad	20,107	0.4379	74,427
u2vcti14cc	20,388	0.41	80,868
u2vcti14cd	20,418	0.4108	80,543
u2vcti14ac	20,563	0.4124	76,890
u2vcti14ad	20,678	0.4166	77,145
lti1cdb1	19,986	0.4129	77,796
lti1adb1	20,489	0.426	73,822
lti180adb1	21,354	0.3815	79,423
geomu24cc	20,001	0.42	76,272
geomu24cd	20,147	0.4176	78,012
geomu24ac	20,038	0.4173	75,274
geomu24ad	19,865	0.406	79,194

Table 6.7: Overview of clusters per session approach with additional features. The table shows the amount of overall clusters, the silhouette score as well as the amount of data points that could not be associated with any cluster..

When adding additional features to the clustering, the results become more diffuse. In another run, the averages for time on site and category per session were added to the feature set. To avoid putting any assumptions into the data distribution, parameters were left as default. The two additional features are intended to add some variety to what are still thought as defining features of the session approaches. As the average number of categories is inherently defined by the logical comparison mechanic, in theory it should be somewhat different when comparing logical to mechanical sessions. The average time on site is assumed to be an even more distinctive feature overall: where the purely logical sessions may go on endlessly, the mechanical sessions are strictly defined by their thresholds. The combined approaches may inherit a bit of both worlds with very short and very long sessions. Evidence of an even stronger effect is expected between the comparison contexts. Table 6.7 shows the results of this clustering approach. Allowing the algorithm to detect even very small clusters strongly effects the outcome, with the number of clusters dramatically increasing and the silhouette score decreasing. The numbers of users assumed to be noise are also higher. This is logical. The algorithm will detect even very small clusters (that may not be very well defined) up to a minimum of five samples, leading to a lower silhouette score and a very high number of clusters. The noise cluster takes that highest share in all the session approaches. Although there are differences among all the session approaches, their results are more or less similar.

When looking at the top 10 clusters per approach again in Table A16, the differences become more visible. While the first three clusters are again almost identical for all session approaches, there are clear differences among the other clusters. The time on site seems to be a highly defining factor; this is very obvious in the logical approaches with no time constraint at all, holding users with a low to medium number of sessions with very high time-on-site averages. The mechanical sessions have far lower values as do the combined approaches, depending on the timeout in use. Another obvious divergence can be seen when comparing the comparison contexts again. Depending on whether the comparison context is consecutive or interleaving, the average values differ greatly. The consecutive variants

[3] Compare `https://hdbscan.readthedocs.io/en/latest/api.html`, retrieved 28 November 2021.

show more similarity to the mechanical sessions, while the variants allowing interleaving behaviour seem to differ in structure.

When looking at the results from a user behaviour-group perspective, it becomes quite obvious that the highest share of users behaves similarly. No matter the session approach, from a structural point of view, these users all behave identically: a small number of sessions with a low number of interactions, spending little time on the system and engaging with only one category. These probably represent users who visit the system only once or twice with little or no connection between visits. Aside from this user group, varying values can be observed throughout. It is important to underline that this variance does not represent actual user behaviour but characterizes the session approach responsible for the appearance of different behaviour.

Only the largest top 10 clusters are displayed: due to the high number of clusters, the shares are low; the clustering is very granular and shows highly specific groups with similar behavioural values. Ultimately, it is likely that the clusters among the session approaches are quite similar when every cluster is compared to each other. These results would be almost useless in a productive environment since, not only are the clusters far too granular but also a total of 20,000 clusters is excessive and too high to be useful. The challenge here would be to find the most appropriate preprocessing pipeline and an algorithm with suitable parameters. Still, the results showing the top 10 clusters in Table 6.7 are meaningful: for example, the finding that the first three clusters are almost always the same (identifying the same behaviour for these users) but aside from these are quite different, is proof enough that the session approaches may produce very different user values.

Overall, it has been shown that the different session approaches can be clustered quite differently. The results seem somewhat similar when comparing the statistics per cluster, but it is highly likely that the actual contents of these clusters look different. A user may end up in completely different clusters depending on the session approach. This can be quite easily observed when comparing the descriptive statistics per user – if they are different across the approaches, the user is likely to end up in different clusters. The extent to which this happens can be analysed relatively straightforwardly with clustered data. When comparing the descriptive statistics per user per cluster and session approach, it should become visible how frequently the identified behaviours of users differ.

The differences are quite high, just as expected: of the roughly 390,000 users in the dataset, only around 8,000 share identical clusters across multiple session approaches when engaging with the four different features. This is a quite strong argument for differing behaviour. Only a fraction of users shows the same behaviour regardless of the session model, allowing for completely different interpretations of their behaviour.

This is a highly interesting observation. In view of the fact that clustering algorithms are used frequently by businesses to identify potentially important user groups (i.e. financially valuable or highly engaged overall) and that marketing strategies are adapted accordingly; the different session approaches have the potential to make a huge impact. Marketing campaigns target users quite differently according to whether they make one session or many. Granted, the choice of features may be too naive and basing a marketing campaign solely on information like this (where a lot of elements are quite arbitrarily chosen) would be reckless to say the least, but the overall impact is clear. Using the results from the different session approaches, system owners utilizing the data can – willingly or unwillingly – make very different statements, thereby increasing or decreasing the system's business performance.

6.4 Measuring the Impact

The final section of this chapter is dedicated to discussing the results from the three use cases. The objective now is to highlight the most important points and to summarize the outcomes of the second part of the evaluation. Having seen the various session approaches deliver fairly diverse results in three quite different use cases, the results are very obvious.

The first use case gave an overview of the structural composition of the different session approaches. In this experiment, an embedding algorithm was trained on the different sequences per session approach to enable it to find category similarities. The underlying business case is simple: with knowledge about category similarity in user sequences it is possible to validate the category tree as the internal page structure of the system. It is beneficial to have similar categories in the same branch of the category tree both to improve navigation and enhance search engine crawlers' ability to crawl the website for their indices. This works only under the assumption that sessions are seen as a construct dealing with the same topic; users will work on exactly one information need during a session. The session-identification approach defines how these sequences will look.

The goal of this use case was to prove how diverse the differently defined sequences are in practice. By looking at the number of distinct root categories per similar categories per **category_id**, it is possible to estimate how topically close the identified categories per category are. In addition, the scores of the cosine similarity per evaluated category can be used to learn about the sequences' structures. Furthermore, it can be assumed that the cosine similarity depends very much on the input data, which indicates the confidence level of the algorithm with regard to category similarity.

Huge variations between the different session approaches were shown. The primary and most important point to consider is the divergence in the numbers of tokens between the different session approaches, which appear to have a correlation with the cosine similarity. The more tokens used as input data, the higher the overall cosine similarity scores (among the top 25 categories, at least). Removing consecutive repeated items in the sequences as well as eliminating sequences with only one interaction have very different effects on the various session definitions. It is obvious that the combined approaches with a) a short timeout or b) the consecutive comparison context have the least tokens but get the highest similarity scores. It can be deduced that this is caused by their internal structure. The sequences from the mechanical examples seem to contain a great diversity of different categories (as underlined by the many tokens), leading to far lower scores, essentially indicating a lesser confidence of the algorithm (under these training circumstances). The purely logical sessions, the geometric sessions and the combined approaches with the interleaving comparison context all allow a higher level of confidence. The lower number of tokens still indicates diversity, but probably one more focused on a fairly similar area of categories – the categories in these sequences more often appear together, hence the higher similarity score. This is again improved by the combined approaches and the consecutive comparison contexts, the most prominent example being the **u2vcti30cd** variant. There is a straightforward reason for the high scores and the low number of tokens. Because the sessions identified by this variant contain a highly specific range of categories, many of the sequences are removed, resulting in a low number of tokens but a highly robust set of category sequences – the same categories appear in the same neighbourhood again and again, allowing the algorithm to identify similarity with high confidence.

The chord chart shown in Figure 6.1 underlines this. Although it depicts a specific category, it exemplifies the obvious different relations of the various approaches. The results are reasonable; considering that the logical sessions are built upon topical similarity created by the user history, it actually makes sense that their results appear to be far more stable and specific than the mechanical variants. The consecutive comparison contexts combined with the 30-minute timeout for the combined sessions is also reasonable; here, the identified sequences are likely to be the most specific, resulting in highly defined topically related category clusters.

This very fact of their specificity may well be why these variants perform best in the recommendation scenario. Using the sessions identified by the different approaches as input data, the algorithm was trained to generate a list of potentially interesting categories for every step in a session. This is a very classical business case: either presenting interesting categories to the user according to their interests or even predicting the potential next click to be able to guide the user towards the fulfilment of a business goal would be a valuable feature for a system. Four different algorithms were tested in order to be able to highlight the specific properties of the input sequences: a most popular item recommendation, a k-nearest Neighbours algorithm and two recurrent neural networks (one purely session-based and the other with a session-representation of previous sessions).

The set-up for this use case was based on the research of Ruocco et al. [225]. The algorithms were trained on 80% of sessions per user and evaluated on the other 20%. Sessions were ordered chronologically based on the timestamp of their first interaction, potentially mixing all sequences of interleaving approaches. In relation to this test set-up, it is important to note that performance of the model is not in theory an indicator of the predictive power of the algorithm due to the different test sets. That said, this dissertation views session identification as part of the experiment set-up and preprocessing, just as it would be treated in a productive environment anyway. The results are still valid, therefore, especially when the differences between the session approaches are considered.

The preprocessing has a quite strong effect. The removal of consecutive repeated items and all users with less than three sessions is a challenging but necessary step in line with the nature of the scenario. Since the experiments for the second use case are similar to those of the first use case, the same effects are to be expected. Despite the divergences both in the nature of the results and their evaluation when comparing use cases one and two, the outcome and the interpretation of them are quite similar. This is perfectly reasonable, since both experiments are based on category sequences, whereas for the recommendation scenario, recall and mean reciprocal rank at different k's were used for evaluation.

To summarize the results, no matter the algorithm the same effects identified for the category embeddings are observed. The mechanical approaches tend to have rather diverse sequences whereas the combined consecutive approaches are highly specific. From a diversity point of view, the geometric approaches appear to be a mixture of both worlds. Best performing are the specific and temporally close sequences of the combined approaches with a consecutive comparison context, which is an extremely reasonable finding. As users' information needs change over time – evolving into different needs, widening or narrowing a topic scope – the recency argument for this type of approach makes a lot of sense. With the additional bonus of highly specific sequences, it seems very logical that these approaches appear to perform best. The time aspect is of central importance apparently: the combination of the topical similarity with a temporal component to symbolize recency improves the quality of recommendations and predictions. Strictly speaking, this is not a new finding: there are comparable recommendation scenarios in the literature that describe work using RNNs aimed at incorporating time between events to improve their recommendations [268, 300]. It seems that adding a time component is critical to the correct attribution of the current information need. The results of the current use case heavily underline this issue, seen by the improvements of the direct comparison compared to the comparisons using all interactions of a previous session to construct logical sessions, thereby attributing a higher importance to the most recent interaction. Arguments as to why the combined approaches using a 30-minute timeout or even the 24-hour timeout seem to deliver the best results across all models and approaches can therefore be put down to a combination of several reasons:

- Recency of category interactions
- Topical proximity of interactions
- Model and system specifics

This dissertation takes the first two reasons as the more important ones, with the tendency of the logical component being the principal element. As this applied recommendation task has shown, the combined approaches with short timeouts apparently are able to produce the best results, although the purely logical variants perform very well also. The concept of the logical sessions (and therefore also the combined approaches) works very well with recommendations. The example from the literature reproduced in this use case tries something similar: to show how incorporating information from previous sessions into session-based recommendations is inherently the same as creating logical sessions – the ideas behind this are very similar, creating topically related context for current interactions.

It is interesting to see that the logical sessions still work better in comparison to the mechanical variants even when using the additional context of the iRNN. For the mechanical sequences, using previous session embeddings should improve the current session-based recommendation to a certain extent. For the logical variants, this is actually counterintuitive, as the sessions already have the similar categories in one sequence. Intuitively, adding previous session context may even dilute the results, but apparently the algorithm is able to differentiate accordingly. This is an interesting finding that hints at the power of these algorithms. As pointed out, the time element is another important addition: the recency effect on the sequences is responsible for highlighting the current context and generating shorter, highly specific sessions.

Finally, and by no means least, the performance of the algorithms is as dependent on the model as it is on the system. For example, the performance of the most popular recommendations seems to be very dependent on the system itself. In view of the fact that a huge portion of interactions lands on the *Smartphones*, it could be that the recommendations may be somewhat disturbed here. However, it is a good baseline approach, which makes it all the more interesting that some approaches still achieve higher values than others here, especially MRR@1. Additionally, the specifics of the model itself are important: In a productive environment, tuning the hyperparameters for the different models according to the specifics of the session approach may lead to improved results.

The third use case delivered results rather different to the first two cases. From an evaluation point of view, the situation is similar to the category embeddings: there is no good or bad result, the use case is simply supposed to demonstrate the differences in the results. Therefore, little effort was spent to find the right feature set or the most suitable preprocessing steps. The different input data was fed into the same algorithms and the results were described. The difference compared to the other use cases manifests itself in the form of the input data. Having previously used sequential **category_id** data at the **session_id** and **user_id** levels, now the clustering was performed on aggregated session data at a **user_id** level. This is quite an important distinction, for, whereas previously the contents of the sessions was the crucial factor, now the algorithm only has the structure of the sessions as input.

The business case is once again very typical in this scenario: clustering the user population in order to gain insights into or new hypotheses on how said users interact with the system. Clearly, it is important for a system to have knowledge about their users: how they behave, how they interact with the system and how they react to customer-related communications. It is only through this that the system owner gains knowledge about user behaviour and the strategy most likely to fulfil certain business goals or incentivize users to help do so.

Using a hierarchical density-based clustering algorithm, two separate runs with different settings and features were conducted. The outcomes were in line with the expectations, showing very distinct results in the clusters associated with users dependent on the session approach. The content of the sessions is not the only element that changes per different identification, the structure changes very much also. Even though the clustering is heavily dependent on both the preprocessing and the features used as well, it became obvious that, depending on which session approach is used to identify the sessions, user behaviour is represented differently.

Where the first two use cases show that the tested session approaches perform very differently under the given circumstances, the third use case shows an additional component. It shows that while the clustering does not reveal at all which session approach delivers better results, it does very clearly prove that users will be clustered differently depending on the session approach. What this means is, if a system seeks to evaluate user behaviour based on session behaviour to find the most engaged users or those most likely to generate the highest financial value, the choice of session approach will be responsible for different results. This is a very important observation since, basically, it means that the system should be optimized to meet different objectives depending on the identified session. The varying identification of sessions leads to varying interpretations of user behaviour on the system.

This concludes the evaluation chapter of this dissertation. Here, it has been shown that there are quite some differences not only in the content but also in the structural composition of the sessions. All three use cases showed another aspect of the data and how sessionized data can be applied in a business setting. As discussed, the session approaches make room for divergent interpretations that lead to potentially very different conclusions for system owners.

Chapter 7

Conclusion

The final chapter of this dissertation provides a brief summary of the research and its most important findings. In addition to discussing the potential limitations and shortcomings of the research, a brief outline of possible starting points for future work is offered.

7.1 Summary

The dissertation opened with a thorough overview of the various concepts of sessions in web information systems. It then went on to investigate the overarching research question as to whether different session-identification implementations would impact analysis and machine learning tasks and to survey the following three subquestions:

RQ1 How can sessions be modelled to represent one or more information needs?

RQ2 Are there differences in the results of different session-identification algorithms? Can these results be attributed to specific identification and comparison mechanics?

RQ3 Will the performance of machine learning algorithms change depending on the input data?

Chapter 2, the literature review, introduced the various concepts of sessionization and the fairly long history of session-identification research. The state of research was outlined from the start: from 1995 with the introduction of simple mechanical sessions and the probable origin of the well-known industry standard – the 30-minute inactivity timeout – by Catledge and Pitkow [43] to the rise of task-focused logical concepts of around 2010 with Jones and Klinkner [120], Gayo-Avello [80] or Hagen et al. [89] [88] and Liao et al. [145]. The different concepts and their respective mechanics were presented and explained with various examples from the literature.

The research has shown how the commonly used evaluation methods are somewhat flawed and impractical to achieve the objective assessment of session-identification approaches. A brief introduction to how sessions are used in online information systems was also given. State-of-the-art examples from the literature were then described to highlight how these might benefit from other session concepts.

Chapter 3 introduced the information system that was used for the research experiments. The system architecture, its features and structure and the most common business cases were explained. In addition, the tracking concept active on the system was described along with the problems that could possibly arise due to its implementation.

Chapter 4 presented the research design, explained the concepts and definitions, and put forward the concise terminology established in this dissertation. It was shown how the the dataset used in the research experiments was preprocessed to a final state, ready to be further enriched with session information. Section 4.4 described in detail how all 135 session approaches tested in this dissertation were constructed and implemented. The section explained all the steps in the development, from the simple mechanical timeouts and fixed lengths towards the more complex logical sessions using category similarity constructed from category sequence embeddings. A novel approach tested in this dissertation was presented: basing the calculation of category similarity and the identification of logical sessions on user-category interaction history using embedding algorithms. The section showed the methodological steps to reproduce these logical sessions using real data from an online information system, thereby answering RQ1. Also described is implementation of the adapted geometric session-identification approach reported by Gayo-Avello [80], whereby instead of using lexical query similarity semantic category similarity was used.

Finally, the steps of the two-part evaluation were presented: refraining from using a traditional gold-standard dataset measuring the actual quality of the identified sessions, a different approach for evaluation was chosen instead. To answer RQ2, the dissertation determined that using a thorough descriptive analysis in the first step of the evaluation was the correct choice to show the divergences between all 135 implemented session variants. The analysis was not aimed at assessing the quality of the produced results but simply at highlighting the differences. The evaluation's second part was more practical: it used a sample of sessions from a selection of 26 approaches as input for machine learning algorithms in the implementation of three typical business applications. The objective of this step was to show the differences between the session-identification algorithms in a less abstract way and to indicate the potential qualitative differences between them depending on the application.

Chapter 5 was dedicated to the initial evaluation; providing extensive analysis based on a variety of important measures, the differences between the results of the session algorithms were elaborated to answer RQ2. An overview of the overall structure of the dataset with a short user analysis was then presented, followed by sections that discussed all types of tested session approaches in detail. This was followed by a discussion that put together the most important results. These results are summarized in Table 7.1. The table lists some of the more salient strengths and weaknesses and provides some general comments on the different approaches, which it attempts to set out in relation to each other.

Approach	Strengths	Weaknesses	Comments
Visits Visits are important for any system. The ability to understand how users navigate an information system and in which order pages are looked at is invaluable information. Only by using visits it is possible to actually understand interleaving behaviour or multitasking with regard to system usage.	• replicate actual user behaviour • path analysis enhances understanding of the user's path through the system • feature adoption can be analysed in the light of user behaviour	• completely dependent on data quality • rather complex calculation with high error potential	• visits should be seen at best as an additional session concept • suited for understanding system navigation • because of calculation complexity not necessarily useful for production applications
Mechanical Timeouts The timeout sessions, especially the inactivity timeout variants, are the industry standard and not without reason. Although their basic assumptions on time affecting the information are highly domain dependent, they serve as a good measure of regular behaviour on a system.	• easy to calculate: only identifier and timestamp are needed • easy to comprehend the technical concept • are uniform across systems and may be used as a standard for across-system comparison • best suited for observing regular and recurring behaviour of users / system performance within a set time frame	• independent from information needs; they simply aggregate interactions over time with no regard for content • there is no definite proof for a global rule that fits all users • users may behave differently – any non-user focused time-related measure will treat all users the same, leading to falsely identified sessions (depending on the assumption)	• inactivity timeout and maximum length may identify the same sessions depending on the system • a global timeout may exist, but it is also domain-dependent • a timeout analysis can give insights into global user behaviour and should be a precondition for mechanical sessions • changing the timeout has a noticeable impact on the resulting sessions
Topical Similarity Topical (logical) similarity is a highly domain-dependent way of understanding the engagement of users with the contents of the system. Depending on the implementation (i.e. comparison context and comparison method), these sessions can create very precise topical sequences, longer-lasting shopping processes or even broad journeys with multiple subtasks.	• allows analysis of user engagement with specific topics over a long period of time • highly adaptable to any scenario: content embeddings are able to extract broad journeys or very specific tasks with limited scope depending on comparison methods and comparison contexts; lexical matching (as implemented here) is very broad, bm25 shared-term space seems rather narrow • analysis of these sessions may reveal how certain topics are approached	• rather complex to implement • many different comparison methods and comparison contexts may seem overwhelming and hard to understand • finding the correct level of similarity between contents may be a long analytical process • more dependent on data quality than mechanical approaches	• best suited for specific topic analysis over time • ideal as an addition to mechanical approaches to understand user behaviour regarding specific content • well suited for a variety of applications (i.e. determining sublevels of similarity) • comparison contexts have a stronger impact than comparison method and should be chosen according to the use case / system
Combined Approaches Subjectively, a combination of logical and mechanical boundaries is the best way to incorporate sessions in applications. They utilize the strengths of both types, potentially leading to improved performance of algorithms. The choice of factors to combine is highly domain-dependent and should be taken with care.	• highly adaptable and flexible • even more versatile than the purely logical approaches as they offer another dimension to adapt scope of identified sessions: comparison method, comparison context, temporal boundary • combination of strengths to create highly specific session approaches	• rather complex to implement and model • finding the correct scope and choice of dimensions (comparison method, comparison context, temporal boundary) can be challenging • the number of dimensions to take into account may be hard to adjust because they cross-interact	• best suited for many different applications of sessions • show strong improvements in, for example, recommendation tasks compared to mechanical approaches • the geometric sessions seem to be a good balance of mechanical and logical boundaries: both dimensions have a degrading nature, making them viable for many systems • the different dimensions show varying strength of impact

Table 7.1: Summary of strengths and weaknesses of session approaches.

Chapter 6 then presented in detail how the sampled input data from 26 algorithms is fed into three machine learning algorithms to answer RQ3. Experiments were run implementing three different business cases: an embedding scenario to calculate category similarity comparable to the procedure from Section 4.4.3.3, a recommendation scenario with multiple different algorithms reproduced from the literature [225] and a clustering use case using a well-suited clustering algorithm [34] [172] for exploratory data analysis. This dissertation demonstrates that the output of these three algorithms changes drastically in some cases and in some experimental circumstances with the data used here (RQ3).

With the design and development of the 135 session models, the extensive descriptive analysis and implementation in three typical business case applications, this dissertation shows in detail that selecting a suitable session model is highly important. It demonstrates how when sessions are identified differently they may deliver varying interpretations of user behaviour depending on how these users interact with the system. This dissertation has underlined the fact that the same divergences are seen in the content also and that the differences are therefore not only structurally superficial. This leads to the very likely assumption that there are not only structural changes but also qualitative differences: using the best session algorithm for certain applications will definitely lead to improved results. In this regard, the dissertation concludes that sessionization should not be treated as a given fact, because careful modelling of sessions is integral to the preprocessing of data.

7.2 Contributions

In its conclusions, this dissertation has reported several findings to support its initial statement: the research has found that the choice made between mechanical, logical or combined session approaches for sessionizing data has a strong impact on the output. Equally, it has been proven that the comparison mechanics, comparison contexts and their subvariants make a difference to how the data may be interpreted. This is the most important outcome: that the comparison mechanics, contexts and subvariants do make a difference to the system and algorithm, not only in performance but financially as well in the end. In addition, the research can report several other findings that will contribute to future research and practice.

1. The establishment of a concise terminology for all session concepts, their subvariants and their underlying logics enables future researchers to work with a common language and common concepts.

2. The introduction of a method to identify logically connected sessions that represent the information needs of users. Thus, instead of using queries, the topical connection, based on user sequence history, calculates category similarities to estimate interaction similarity. This mechanic enables system owners to understand the engagement and needs of its users when interacting with a certain topic area without relying solely on queries. The algorithm concept allows different levels of complexity or attachment,

essentially offering the possibility to identify either specific tasks or broad journeys as defined in Section 4.1.

3. Providing a comprehensive and holistic overview of the topic of session-identification algorithms, including implementation, analysis and application of multiple different mechanics, contexts and algorithms.

4. Presenting researchers the possibility to create specific topic-focused session datasets based on actual data: for example, with the logical mechanisms, datasets can be selected based on user sessions which focus on different information needs with varying complexity and scope.

5. Showcasing a methodology with which to evaluate different session concepts in terms of structure and objective quality.

In all, to the best knowledge of the author, the main contribution of the current dissertation is to provide the most comprehensive comparison of session-identification algorithms to date. This research provides a methodology to comprehensively implement, analyse and compare a wide variety of mechanics, making it possible to understand user behaviour from manifold perspectives and allowing system owners to better understand the effects their session modelling has.

7.3 Limitations

Naturally, the dissertation has some limitations. There are multiple caveats in every section that readers should bear in mind when considering the results and contributions. The limitations concern different areas, but the list below contains the overarching topics:

- Data quality regarding the raw data and the assumed concepts
- Issues regarding the session data and preprocessing
- Variety of mechanics and applications
- Methodological limitations

The first thing to consider is the lack of data quality in the raw data. This was already somewhat discussed in Section 3.5 and mentioned during the preprocessing steps in Section 4.3. There are multiple limitations regarding the used dataset that have a direct influence on the identified sessions.

The most fundamental issue is the common user identifier. This dissertation employed a **user_id** concept based on the **cookie_value** and pre-mapped hashed email information. The principle underlying this is that multiple cookies which share common hashes are associated with the last hash seen in the data, creating a unique and anonymous **user_id** for every set of related cookies and hashes. However, this concept may produce too many errors: not least, the association may be faulty – the connection between the last known hash and related cookies is only an assumption. The second point is that the tracking quality itself, based on **cookie_value** as the common user identifier, might not be good enough. It can be taken as read that many system users will not be recognized as they enter the system; that is, users may be employing masking software or simply deleting their cookies regularly / after every visit to the system. This known limitation cannot be fixed by preprocessing the data any differently; when a user chooses not to be tracked, there is no valid technical, legal or ethical option that can override this choice. Unfortunately, it is not possible to estimate the impact of this, but it raises the question of whether the majority of users with a low number of overall interactions do in fact return more than once or twice but with different identifiers.

This research limitation may have had a big impact on the outcome of the tested session algorithms. The absence of a reliable user identifier may well have interfered with any logical consistency, leading to falsely connected or no interaction sequences connected at all. Nonetheless, it is a limitation that cannot be changed. The assumption is that the sheer quantity of data may have somewhat mitigated the effects. In addition, bearing in mind that the effects would have been equal across all session approaches, the differences between them therefore can still be seen as valid, even if the logical sessions may have been hit the hardest. Also, in defence, it should be noted that other published research deals with the exact same challenge and no practical solution has been forthcoming as yet.

Another common problem is information that is not available in the raw data. In the the **url** and **http_referer** fields, for example. In theory, both should have content on every interaction the user has with the system, reporting the way users move on the system. Unfortunately, often they are empty or plainly wrong. The tracking apparently has issues with some elements of the website: sometimes, marketing parameters are added or removed more or less arbitrarily, changing the relation between **url** and **http_referer**. This can be somewhat caught by preprocessing but often there are breaks in the path. An additional problem here are completely missing traces or completely untracked content (i.e. areas behind the login including the wish list). Both present very significant problems that again have potential to lead to inconsistencies. For example, the research showed how the visits depend very much on data quality in this regard. The many session breaks and inconsistent sessions (and the highest number of single-interaction sessions) produced by the path-based approach made it essentially useless.

A similar limitation revolves around this same issue of data quality. As Section 4.4.3.2 discussed, many interactions needed to be substituted with a **category_id** to make use of them in the logical sessions. This is a) a problem of the raw data, not containing reliable identifiers although it should and b) a known limitation of this research. By adding **category_ids** to the query interactions using word embeddings, a certain error margin was introduced. Once again, the effect of incorrectly assigned **category_ids** is not easily measurable, especially since these issues with identifiers are something to be expected in the raw data. But it is still a limitation that may have had an impact on the logical and combined sessions, potentially interrupting logically connected sequences. However, some session approaches were more reliant on data quality than others.

Generally, many steps in this dissertation relied on heavy preprocessing to generate a usable dataset; many of these came with assumptions that could not really be tested in any detail within the scope of this research. This meant that multiple intermediate results were not actually evaluated in a proper sense, but rather simply regarded as good enough for the time being and under the specific circumstances of the experiments (i.e. the assignment of categories to queries, the thresholds for the logical sessions, the sampling). This limitation belongs within the scope of this dissertation, therefore; in order to show the differences between the session algorithms, the reported results were good enough. In the future, the research and even practical implementation of the steps shown here should be evaluated with care even though they seem ‘good enough’ for now.

Overall, the 135 variants, tested in three overarching categories with 11 subcategories, brought an impressive variety and introduced many different mechanics and contexts. However, there is still room for more: the logical sessions used only three different comparison mechanics (lexical matching, bm25 shared-term space and user sequence embeddings). Greater diversity could definitely be introduced in relation to these mechanics: for example, using a variety of distance calculations of the similarity may lead to different results; or even completely different similarity mechanics using other features may produce worthwhile findings. The same is true for the combined approaches using only one example from the literature: extended diversity would have been beneficial to produce an even broader picture. These limitations were considered acceptable within the scope of this dissertation, whereby a limited diversity of mechanics did not invalidate the assertion that there are clear divergences between the different mechanics.

Additionally, and this relates also to the previous point, the dataset used in this research is quite distinct. As it contains a limited number of query interactions, it is near-impossible to transfer any algorithm used in other research for session identification directly to this dataset. The majority of the state-of-the-art research deals with query-based search interaction datasets, usually only calculating similarity between queries. These algorithms are not directly applicable on data structured in other ways. The point presents both a limitation and an advantage in its contribution: the testing of these session algorithms on a dataset that is not truly comparable to the other literature, also demonstrates a way that research in this area and on similar datasets can be carried out.

Likewise, another methodological limitation is the difference in evaluation. While it is commonplace to use a gold standard to evaluate session algorithms, this dissertation disregarded this type of evaluation since it is prone to introducing subjectivity. The current research, therefore, evaluated and tested the differences in an alternative way. For example, an application from the literature was incorporated and tested and was shown to reproduce the same limitations. As Ludewig and Jannach [156] report in their evaluation of many different session-based recommendation scenarios, without a standard for evaluation the various research is not really comparable anyway. However, again, given the circumstances, the results of the current research experiment are valid.

A final point can be made about the selection of applications. The algorithms chosen for this research can be considered state-of-the art, one testing an example from the literature and the other two utilizing well-tested machine learning models. The applications that were tested delivered meaningful results by demonstrating how differently the session approaches behave under given circumstances. Nonetheless, other more diverse and also other traditional use cases could be recruited to demonstrate that there will always likely be divergences even in something simple like a regression forecast.

7.4 Suggestions for Future Work

There are many possibilities to build upon the research conducted here. There are several parts throughout this dissertation where more thorough evaluation or implementation may have led to improved results. Developing these parts could offer a good starting point for further research. There are some other ideas to present for future research based on the findings reported in this dissertation.

The first of these that immediately springs to mind is to simply repeat the research conducted here on different datasets. This dissertation has proven that there is a divergence in the output of different session modelling in relation to interactions on an online price comparison platform – a rather specific type of online information system. A no doubt fruitful continuation in this line of work would be to test the methodology presented here, simply reproducing it on different data to see if the statements hold generally true.

A further interesting step forward would be to develop more sophisticated and variegated means to represent information need in logical sessions. Here, there are many options that could be built upon to research finer-grained or broader logical sessions, representing either very specific tasks or holistic user journeys. This dissertation presents both, depending on the chosen method; future work might expand on this and create a framework with different thresholds or completely other mechanics. A framework of hierarchies of logical sessions – representing one or more information needs depending on mechanic and threshold – would be an interesting project, enabling researchers to identify nuances in outcome between the logical and combined sessions: presenting a way of programmatically evaluating user satisfaction and users' information needs.

Future research might involve expanding the testing of logical and combined sessions and extending the comparisons between them. There is room to improve the depth and diversity of the use cases. Evaluation of the recommendations could be conducted at greater scale, for example, quantifying the effect of different scales of logical relatedness in the session-based and session-aware recommendations. Likewise, the exploratory clustering conducted in this dissertation could very well be extended with other appropriate and more thorough preprocessing, carefully selected hyperparameters and added features. Another potential extension might focus on the logical sessions to increase understanding of how the resulting behaviours are clustered and, ultimately, how they look comparatively.

Finally, the introduction of logical sessions has increased knowledge about the different strengths and inherent properties of the presented session approaches, opening up many more possibilities relating to user analysis and user support. One of the more sophisticated ideas to come out of the current research is to exploit the representations built by different types of sessions to simulate user behaviour. This could be an advanced user model employing various levels of logical sessions and adapted mechanical sessions: with these different levels of behaviour as represented by the sessions, an algorithm would be able to construct complex user models. With the information generated by these user models, it would then be possible to actually simulate users interacting with the system, thereby easily creating artificial data to enable automatic evaluation of certain system features.

7.5 Closing Words

This dissertation has successfully answered its overarching research question: There is definitely a strong difference between the results when mechanical, logical or combined session identification is employed, no matter the application. The abundance of mechanics and contexts tested here and the divergence in the results indicate that anyone working with user-interaction data should be very cautious when preprocessing it to identify sessions, as the difference in the output could be greater than expected. The dissertation has highlighted how careless adoption of industry standards like the 30-minute inactivity rule may not only significantly reduce the performance of algorithms but also, ultimately, the systems. The key to understanding users and the usage of any information system is finding the optimum session identification for the right application. Depending on said application and the system and its users in general, the session-identification approach may be different from use case to use case.

Convincing the community overnight is probably not possible, which is lamentable considering that multiple research has already shown that recourse to logical sessions rather than a continued reliance on mechanical sessions will improve the results of algorithms. Hence there remains an incomprehensibly large quantity of applied work that simply adopts the industry standard without sufficient forethought. The author is looking forward at least to gradually incorporating the innovative and improved concepts presented in this dissertation into practice within the industry.

Bibliography

[1] Adaji, I., K. Oyibo, and J. Vassileva (2018). "Shopper Types and the Influence of Persuasive Strategies in E-Commerce". In: *Proceedings of the Third International Workshop on Personalization in Persuasive Technology.* Third International Workshop on Personalization in Persuasive Technology (PPT18), Waterloo, ON, Canada, ed. by R. Orji, M. Kaptein, J. Ham, et al. PPT '18. CEUR Workshop Proceedings, Vol. 2089. CEUR-WS.org, pp. 58–65.

[2] Adomavicius, G. and A. Tuzhilin (2005). "Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions". In: *IEEE Transactions on Knowledge and Data Engineering* 17.6, pp. 734–749. DOI: 10.1109/TKDE.2005.99.

[3] Agichtein, E., E. Brill, and S. Dumais (2018). "Improving Web Search Ranking by Incorporating User Behavior Information". In: *ACM SIGIR Forum* 52.2, pp. 11–18. DOI: 10.1145/3308774.3308778.

[4] Agichtein, E., R. White, S. Dumais, and P. Bennett (2012). "Search, Interrupted: Understanding and Predicting Search Task Continuation". In: *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval.* 35th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR12), Portland, OR, USA, ed. by W. Hersh, J. Callan, Y. Maarek, et al. SIGIR '12. Association for Computing Machinery, pp. 315–324. DOI: 10.1145/2348283.2348328.

[5] Agosti, M. and G. M. Di Nunzio (2007). "Web Log Mining: A Study of User Sessions". In: *Proceedings of the 10th DELOS Thematic Workshop on Personalized Access, Profile Management, and Context Awareness in Digital Libraries.* 10th DELOS Thematic Workshop on Personalized Access, Profile Management, and Context Awareness in Digital Libraries (PersDL07), Corfu, Greece, ed. by W. Kießling, G. Koutrika, T. Catarci, et al. PersDL '07, pp. 70–74.

[6] Ai, Q., Y. Zhang, K. Bi, and W. B. Croft (2020). "Explainable Product Search with a Dynamic Relation Embedding Model". In: *ACM Transactions on Information Systems* 38.1, pp. 1–29. DOI: 10.1145/3361738.

[7] Alexopoulou, P. (2016). "A New Integrated Model for Multitasking during Web Searching". PhD thesis. Loughborough, England, UK: Loughborough University. 259 pp.

[8] Anderson, B. and D. McGrew (2017). "OS Fingerprinting: New Techniques and a Study of Information Gain and Obfuscation". In: *Proceedings of the 2017 IEEE Conference*

on Communications and Network Security. 2017 IEEE Conference on Communications and Network Security (CNS17), Las Vegas, NV, USA. CNS '17. IEEE, pp. 1–9. DOI: 10.1109/CNS.2017.8228647.

[9] Ansari, Z. A., S. A. Sattar, and A. V. Babu (2017). "A Fuzzy Neural Network Based Framework to Discover User Access Patterns from Web Log Data". In: *Advances in Data Analysis and Classification* 11.3, pp. 519–546. DOI: 10.1007/s11634-015-0228-4.

[10] Asadianfam, S. and M. Mohammadi (2014). "Identify Navigational Patterns of Web Users". In: *International Journal of Computer-Aided Technologies (IJCAx)* 1.1, pp. 1–8.

[11] Aslanyan, G., A. Mandal, P. S. Kumar, A. Jaiswal, and M. R. Kannadasan (2020). "Personalized Ranking in eCommerce Search". In: *Proceedings of The World Wide Web Conference 2020*. The World Wide Web Conference 2020 (WWW20), Taipei, Taiwan, ed. by A. E. F. Seghrouchni, G. Sukthankar, T.-Y. Liu, et al. WWW '20. Association for Computing Machinery, pp. 96–97. DOI: 10.1145/3366424.3382715.

[12] Aswadallah, A. H., R. W. White, P. Pantel, S. T. Dumais, and Y.-M. Wang (2014). "Supporting Complex Search Tasks". In: *Proceedings of the 23rd ACM International Conference on Information and Knowledge Management*. 23rd ACM International Conference on Information and Knowledge Management (CIKM14), Shanghai, China, ed. by J. Li, X. S. Wang, M. Garofalakis, et al. CIKM '14. Association for Computing Machinery, pp. 829–838. DOI: 10.1145/2661829.2661912.

[13] Aula, A., R. M. Khan, and Z. Guan (2010). "How Does Search Behavior Change as Search Becomes More Difficult?" In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. SIGCHI Conference on Human Factors in Computing Systems (CHI10), Atlanta, GA, USA, ed. by E. Mynatt, G. Fitzpatrick, K. Edwards, et al. CHI '10. Association for Computing Machinery, pp. 35–44. DOI: 10.1145/1753326.1753333.

[14] Bandari, D., S. Xiang, J. Martin, and J. Leskovec (2019). "Categorizing User Sessions at Pinterest". In: *Proceedings of the 2019 IEEE International Conference on Big Data and Smart Computing*. 2019 IEEE International Conference on Big Data and Smart Computing (BigComp), Kyoto, Japan. BigComp '19. IEEE, pp. 1–8. DOI: 10.1109/BIGCOMP.2019.8679211.

[15] Barkan, O., Y. Brumer, and N. Koenigstein (2016). "Modelling Session Activity with Neural Embedding". In: *Proceedings of the Poster Track of the 10th ACM Conference on Recommender Systems*. 10th ACM Conference on Recommender Systems (RecSys16), Boston, MA, USA, ed. by I. Guy and A. Sharma. RecSys '16. CEUR Workshop Proceedings, Vol. 1688. CEUR-WS.

[16] Batmaz, Z., A. Yurekli, A. Bilge, and C. Kaleli (2019). "A Review on Deep Learning for Recommender Systems: Challenges and Remedies". In: *Artificial Intelligence Review* 52.1, pp. 1–37. DOI: 10.1007/s10462-018-9654-y.

[17] Bayir, M. A., I. H. Toroslu, M. Demirbas, and A. Cosar (2012). "Discovering Better Navigation Sequences for the Session Construction Problem". In: *Data & Knowledge Engineering* 73, pp. 58–72. DOI: 10.1016/j.datak.2011.11.005.

[18] Beeferman, D. and A. Berger (2000). "Agglomerative Clustering of a Search Engine Query Log". In: *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD00), Boston, MA, USA, ed. by R. Ramakrishnan, S. Stolfo, R. Bayardo, et al. KDD '00. Association for Computing Machinery, pp. 407–416. DOI: 10.1145/347090.347176.

[19] Belkin, N. J., M. Cole, and J. Liu (2009). "A Model for Evaluation of Interactive Information Retrieval". In: *Proceedings of the SIGIR Workshop on the Future of IR Evaluation.* SIGIR Workshop on the Future of IR Evaluation (SIGIR09), Boston, MA, USA, ed. by S. Geva, J. Kamps, C. Peters, et al. SIGIR '09. IR Publications, pp. 7–8.

[20] Berendt, B., B. Mobasher, M. Nakagawa, and M. Spiliopoulou (2003). "The Impact of Site Structure and User Environment on Session Reconstruction in Web Usage Analysis". In: *Mining Web Data for Discovering Usage Patterns and Profiles: Proceedings of the 4th International Conference on Mining Web Data for Discovering Usage Patterns and Profiles.* International Workshop on Mining Web Data for Discovering Usage Patterns and Profiles (WebKDD02), Edmonton, AB, Canada, ed. by O. R. Zaïane, J. Srivastava, M. Spiliopoulou, et al. WebKDD'02. Lecture Notes in Computer Science, Vol. 2703. Springer, pp. 159–179. DOI: 10.1007/978-3-540-39663-5_10.

[21] Beverly, R. (2004). "A Robust Classifier for Passive TCP/IP Fingerprinting". In: *Passive and Active Network Measurement: Proceedings of the 5th International Workshop on Passive and Active Network Measurement.* 5th International Workshop on Passive and Active Network Measurement (PAM04), Antibes Juan-les-Pins, France, ed. by C. Barakat and I. Pratt. PAM '04. Lecture Notes in Computer Science, Vol. 3015. Springer, pp. 158–167. DOI: 10.1007/978-3-540-24668-8_16.

[22] Bhandari, G., D. Bussiere, and P. Voyer (2018). "How Loud Is the Scream of a Clickstream? Insights from Big Data Analysis". In: *Proceedings of the 24th Americas Conference on Information Systems.* 24th Americas Conference on Information Systems (AMCIS18), New Orleans, LA, USA. AMCIS '18. Association for Information Systems, pp. 1–5.

[23] Bigon, L., G. Cassani, C. Greco, L. Lacasa, M. Pavoni, A. Polonioli, and J. Tagliabue (2019). "Prediction Is Very Hard, Especially about Conversion. Predicting User Purchases from Clickstream Data in Fashion e-Commerce". In: *CoRR.* Arxiv abs/1907.00400. DOI: 10.48550/arXiv.1907.00400.

[24] Bloch, P. H., D. L. Sherrell, and N. M. Ridgway (1986). "Consumer Search: An Extended Framework". In: *Journal of Consumer Research* 13.1, pp. 119–126. DOI: 10.1086/209052.

[25] Boda, K., Á. M. Földes, G. G. Gulyás, and S. Imre (2011). "User Tracking on the Web via Cross-Browser Fingerprinting". In: *Information Security Technology for Applications: Proceedings of the 16th Nordic Conference on Information Security Technology*

for Applications. 16th Nordic Conference on Information Security Technology for Applications (NordSec2011), Tallinn, Estonia, ed. by P. Laud. NordSec '11. Lecture Notes in Computer Science, Vol. 7161. Springer, pp. 31–46. DOI: 10.1007/978-3-642-29615-4_4.

[26] Bogina, V. and T. Kuflik (2017). "Incorporating Dwell Time in Session-Based Recommendations with RNN". In: *Proceedings of the 1st Workshop on Temporal Reasoning in Recommender Systems Co-Located with 11th International Conference on Recommender Systems*. 1st Workshop on Temporal Reasoning in Recommender Systems Co-Located with 11th International Conference on Recommender Systems (RecSys 2017), Como, Italy, ed. by M. Bielikova, V. Bogina, T. Kuflik, et al. RecSys '17. CEUR Workshop Proceedings, Vol. 1922. CEUR-WS.org, pp. 57–59.

[27] Bojanowski, P., E. Grave, A. Joulin, and T. Mikolov (2017). "Enriching Word Vectors with Subword Information". In: *Transactions of the Association for Computational Linguistics* 5, pp. 135–146. DOI: 10.1162/tacl_a_00051.

[28] Boldi, P., F. Bonchi, C. Castillo, D. Donato, A. Gionis, and S. Vigna (2008). "The Query-Flow Graph: Model and Applications". In: *Proceedings of the 17th ACM Conference on Information and Knowledge Management*. 17th ACM Conference on Information and Knowledge Management (CIKM08), Napa Valley, CA, USA, ed. by J. G. Shanahan, S. Amer-Yahia, I. Manolescu, et al. CIKM '08. Association for Computing Machinery, pp. 609–618. DOI: 10.1145/1458082.1458163.

[29] Boughareb, D. and N. Farah (2013). "Identify the User's Information Need Using the Current Search Context". In: *International Journal of Enterprise Information Systems (IJEIS)* 9.4, pp. 28–42. DOI: 10.4018/ijeis.2013100103.

[30] Brenes, D. J. and D. Gayo-Avello (2009). "Stratified Analysis of AOL Query Log". In: *Information Sciences* 179.12, pp. 1844–1858.

[31] Broder, A. (2002). "A Taxonomy of Web Search". In: *ACM SIGIR Forum* 36.2, pp. 3–10. DOI: 10.1145/792550.792552.

[32] Brost, B., R. Mehrotra, and T. Jehan (2019). "The Music Streaming Sessions Dataset". In: *Proceedings of The World Wide Web Conference 2019*. The World Wide Web Conference (WWW19), San Francisco, CA, USA, ed. by L. Liu and R. White. WWW '19. Association for Computing Machinery, pp. 2594–2600. DOI: 10.1145/3308558.3313641.

[33] Buzikashvili, N. and B. J. Jansen (2006). "Limits of the Web Log Analysis Artifacts". In: *Workshop on Logging Traces of Web Activity: The Mechanics of Data Collection at the 15th International World Wide Web Conference*. 15th International World Wide Web Conference (WWW 2006), Edinburgh, Scotland, UK. WWW '06.

[34] Campello, R. J. G. B., D. Moulavi, and J. Sander (2013). "Density-Based Clustering Based on Hierarchical Density Estimates". In: *Advances in Knowledge Discovery and Data Mining Part II: Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD13), Gold Coast, QLD, Australia, ed. by J. Pei, V. S. Tseng, L. Cao,

et al. PAKDD '13. Lecture Notes in Computer Science, Vol. 7819. Springer, pp. 160–172. DOI: 10.1007/978-3-642-37456-2_14.

[35] Cao, H., D. Hu, D. Shen, D. Jiang, J.-T. Sun, E. Chen, and Q. Yang (2009a). "Context-Aware Query Classification". In: *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval.* 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR09), Boston, MA, USA, ed. by J. Allan, J. Aslam, M. Sanderson, et al. SIGIR '09. Association for Computing Machinery, pp. 3–10. DOI: 10.1145/1571941.1571945.

[36] Cao, H., D. Jiang, J. Pei, E. Chen, and H. Li (2009b). "Towards Context-Aware Search by Learning a Very Large Variable Length Hidden Markov Model from Search Logs". In: *Proceedings of the 18th International Conference on World Wide Web.* 18th International Conference on World Wide Web (WWW09), Madrid, Spain, ed. by J. Quemada, G. León, Y. Mareek, et al. WWW '09. Association for Computing Machinery, pp. 191–200. DOI: 10.1145/1526709.1526736.

[37] Cao, H., D. Jiang, J. Pei, Q. He, Z. Liao, E. Chen, and H. Li (2008). "Context-Aware Query Suggestion by Mining Click-through and Session Data". In: *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD08), Las Vegas, NV, USA, ed. by Y. Li, B. Liu, and S. Sarawagi. KDD '08. Association for Computing Machinery, pp. 875–883. DOI: 10.1145/1401890.1401995.

[38] Cao, Y., S. Li, and E. Wijmans (2017). "(Cross-)Browser Fingerprinting via OS and Hardware Level Features". In: *Proceedings of the 2017 Network and Distributed System Security Symposium.* Network and Distributed System Security Symposium (NDSS17), San Diego, CA, USA. NDSS '17. Internet Society. DOI: 10.14722/ndss.2017.23152.

[39] Carterette, B., P. Clough, M. Hall, E. Kanoulas, and M. Sanderson (2016). "Evaluating Retrieval over Sessions: The TREC Session Track 2011-2014". In: *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval.* 39th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR16), Pisa, Italy, ed. by R. Perego, F. Sebastiani, J. Aslam, et al. SIGIR '16. Association for Computing Machinery, pp. 685–688. DOI: 10.1145/2911451.2914675.

[40] Caselles-Dupré, H., F. Lesaint, and J. Royo-Letelier (2018). "Word2vec Applied to Recommendation: Hyperparameters Matter". In: *Proceedings of the 12th ACM Conference on Recommender Systems.* 12th ACM Conference on Recommender Systems (RecSys18), Vancouver, BC, Canada, ed. by S. Pera, M. Ekstrand, X. Amatriain, et al. RecSys '18. Association for Computing Machinery, pp. 352–356. DOI: 10.1145/3240323.3240377.

[41] Castagnos, S., A. L'Huillier, and A. Boyer (2015). "Toward a Robust Diversity-Based Model to Detect Changes of Context". In: *Proceedings of the 27th International Conference on Tools with Artificial Intelligence.* 27th International Conference on Tools with Artificial Intelligence (ICTAI15), Vietri Sul Mare, Italy, ed. by L. Tsoukalas, G. Papadopoulos, and A. Esposito. ICTAI '15. IEEE, pp. 534–541. DOI: 10.1109/ICTAI.2015.84.

[42] Castellano, G., A. Fanelli, and M. A. Torsello (2007). “LODAP: A Log Data Preprocessor for Mining Web Browsing Patterns”. In: *Proceedings of the 6th WSEAS International Conference on Artificial Intelligence, Knowledge Engineering and Data Bases - Volume 6*. 6th WSEAS International Conference on Artificial Intelligence, Knowledge Engineering and Data Bases (AIKED07), Corfu, Greece, ed. by C. A. Long, V. M. Mladenov, and Z. Bojkovic. AIKED '07. World Scientific and Engineering Academy and Society, pp. 12–17.

[43] Catledge, L. D. and J. E. Pitkow (1995). “Characterizing Browsing Strategies in the World-Wide Web”. In: *Computer Networks and ISDN Systems* 27.6, pp. 1065–1073. DOI: 10.1016/0169-7552(95)00043-7.

[44] Chen, J., J. Mao, Y. Liu, Z. Min, and S. Ma (2019). “TianGong-ST: A New Dataset with Large-scale Refined Real-world Web Search Sessions”. In: *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. 28th ACM International Conference on Information and Knowledge Management (CIKM19), Beijing, China, ed. by W. Zhu, D. Tao, X. Cheng, et al. CIKM '19. Association for Computing Machinery, pp. 2485–2488. DOI: 10.1145/3357384.3358158.

[45] Chen, M.-S., J. S. Park, and P. S. Yu (1998). “Efficient Data Mining for Path Traversal Patterns”. In: *IEEE Transactions on Knowledge and Data Engineering* 10.2, pp. 209–221. DOI: 10.1109/69.683753.

[46] Cheng, J., C. Lo, and J. Leskovec (2017). “Predicting Intent Using Activity Logs: How Goal Specificity and Temporal Range Affect User Behavior”. In: *Proceedings of the 26th International Conference on World Wide Web*. 26th International Conference on World Wide Web (WWW17), Perth, WA, Australia, ed. by R. Barrett, R. Cummings, E. Agichtein, et al. WWW '17. Association for Computing Machinery, pp. 593–601. DOI: 10.1145/3041021.3054198.

[47] Cheng, Z., B. Gao, and T.-Y. Liu (2010). “Actively Predicting Diverse Search Intent from User Browsing Behaviors”. In: *Proceedings of the 19th International Conference on World Wide Web*. 19th International Conference on World Wide Web (WWW10), Raleigh, NC, USA, ed. by M. Rappa, P. Jones, J. Freire, et al. WWW '10. Association for Computing Machinery, pp. 221–230. DOI: 10.1145/1772690.1772714.

[48] Cherniak, A. and J. Bridgewater (2013). “Session Modeling to Predict Online Buyer Behavior”. In: *Proceedings of the 2013 Workshop on Data-driven User Behavioral Modelling and Mining from Social Media*. 2013 Workshop on Data-Driven User Behavioral Modelling and Mining from Social Media (DUBMOD13), Co-Located with 22nd ACM Conference on Information and Knowledge Management (CIKM13), San Francisco, CA, USA, ed. by J. Mahmud, J. Caverlee, J. Nichols, et al. CIKM '13. Association for Computing Machinery, pp. 1–4. DOI: 10.1145/2513577.2513583.

[49] Chierichetti, F., R. Kumar, P. Raghavan, and T. Sarlos (2012). “Are Web Users Really Markovian?” In: *Proceedings of the 21st International Conference on World Wide Web*. 21st International Conference on World Wide Web (WWW12), Lyon, France, ed. by A. Mille, F. Gandon, J. Misselis, et al. WWW '12. Association for Computing Machinery, pp. 609–618. DOI: 10.1145/2187836.2187919.

[50] Chitra, S. and B. Kalpana (2013). "A Novel Preprocessing Mixed Ancestral Graph Technique for Session Construction". In: *Proceedings of the 2013 International Conference on Computer Communication and Informatics.* 2013 International Conference on Computer Communication and Informatics, Coimbatore, India. IEEE, pp. 1–7. DOI: 10.1109/ICCCI.2013.6466161.

[51] Chitraa, V. and D. A. S. Thanamani (2011). "A Novel Technique for Sessions Identification in Web Usage Mining Preprocessing". In: *International Journal of Computer Applications* 34.9, pp. 23–27. DOI: 10.5120/4127-5958.

[52] Chuklin, A., P. Serdyukov, and M. De Rijke (2013). "Using Intent Information to Model User Behavior in Diversified Search". In: *Advances in Information Retrieval: Proceedings of the 35th European Conference on Advances in Information Retrieval.* 35th European Conference on Advances in Information Retrieval (ECIR 2013), Moscow, Russia, ed. by P. Serdyukov, P. Braslavski, S. O. Kuznetsov, et al. ECIR '13. Lecture Notes in Computer Science, Vol. 7814. Springer, pp. 1–13. DOI: 10.1007/978-3-642-36973-5_1.

[53] Cochran, W. G. (1963). *Sampling Techniques. 2nd Edition.* John Wiley & Sons. 413 pp.

[54] Collins-Thompson, K., P. N. Bennett, R. W. White, S. de la Chica, and D. Sontag (2011). "Personalizing Web Search Results by Reading Level". In: *Proceedings of the 20th ACM International Conference on Information and Knowledge Management.* 20th ACM International Conference on Information and Knowledge Management (CIKM11), Glasgow, Scotland, UK, ed. by B. Berendt, A. de Vries, W. Fan, et al. CIKM '11. Association for Computing Machinery, pp. 403–412. DOI: 10.1145/2063576.2063639.

[55] Cooley, R., B. Mobasher, and J. Srivastava (1999). "Data Preparation for Mining World Wide Web Browsing Patterns". In: *Knowledge and Information Systems* 1.1, pp. 5–32. DOI: 10.1007/BF03325089.

[56] Cui, H., Ji-Rong Wen, Jian-Yun Nie, and Wei-Ying Ma (2003). "Query Expansion by Mining User Logs". In: *IEEE Transactions on Knowledge and Data Engineering* 15.4, pp. 829–839. DOI: 10.1109/TKDE.2003.1209002.

[57] Cui, H., J.-R. Wen, J.-Y. Nie, and W.-Y. Ma (2002). "Probabilistic Query Expansion Using Query Logs". In: *Proceedings of the 11th International Conference on World Wide Web.* 11th International Conference on World Wide Web (WWW02), Honolulu, HI, USA, ed. by D. Lassner, D. De Roure, and A. Iyengar. WWW '02. Association for Computing Machinery, pp. 325–332. DOI: 10.1145/511446.511489.

[58] Daoud, M., L.-T. Lechani, and M. Boughanem (2009). "Towards a Graph-Based User Profile Modeling for a Session-Based Personalized Search". In: *Knowledge and Information Systems* 21.3, pp. 365–398. DOI: 10.1007/s10115-009-0232-0.

[59] Daoud, M., L. Tamine-Lechani, M. Boughanem, and B. Chebaro (2009). "A Session Based Personalized Search Using an Ontological User Profile". In: *Proceedings of the 24th ACM Symposium on Applied Computing.* 24th ACM Symposium on Applied Computing (SAC09), Honolulu, HI, USA, ed. by S. Y. Shin and S. Ossowski. SAC '09. Association for Computing Machinery, pp. 1732–1736. DOI: 10.1145/1529282.1529670.

[60] Dehghani, M., S. Rothe, E. Alfonseca, and P. Fleury (2017). “Learning to Attend, Copy, and Generate for Session-Based Query Suggestion”. In: *Proceedings of the 26th ACM Conference on Information and Knowledge Management.* 26th ACM Conference on Information and Knowledge Management (CIKM17), Singapore, Singapore, ed. by E.-P. Lim and M. Winslett. CIKM ’17. Association for Computing Machinery, pp. 1747–1756. DOI: 10.1145/3132847.3133010.

[61] De Leoni, M. and S. Dündar (2020). “Event-Log Abstraction Using Batch Session Identification and Clustering”. In: *Proceedings of the 35th ACM Symposium on Applied Computing.* 35th ACM Symposium on Applied Computing (SAC20), Brno, Czech Republic, ed. by C.-H. Hung, T. Cerny, D. Shin, et al. SAC ’20. Association for Computing Machinery, pp. 36–44. DOI: 10.1145/3341105.3373861.

[62] Dietz, F. (2020). “The Curious Case of Session Identification”. In: *Experimental IR Meets Multilinguality, Multimodality, and Interaction: Proceedings of the 11th International Conference of the CLEF Association (CLEF 2020).* 11th International Conference of the Cross-Language Evaluation Forum for European Languages (CLEF20), Thessaloniki, Greece, ed. by A. Arampatzis, E. Kanoulas, T. Tsikrika, et al. CLEF ’20. Lecture Notes in Computer Science, Vol. 12260. Springer, pp. 69–74. DOI: 10.1007/978-3-030-58219-7_6.

[63] Ding, A. W., S. Li, and P. Chatterjee (2015). “Learning User Real-Time Intent for Optimal Dynamic Web Page Transformation”. In: *Information Systems Research* 26.2, pp. 339–359. DOI: 10.1287/isre.2015.0568.

[64] Dinuca, C. E. and D. Ciobanu (2012). “Improving the Session Identification Using the Mean Time”. In: *International Journal of Mathematical Models and Methods in Applied Sciences* 6.2, pp. 265–272.

[65] Dixit, V. S. and S. K. Bhatia (2015). “Refinement and Evaluation of Web Session Cluster Quality”. In: *International Journal of System Assurance Engineering and Management* 6.4, pp. 373–389. DOI: 10.1007/s13198-014-0266-x.

[66] Domenech, J. M. and J. Lorenzo (2007). “A Tool for Web Usage Mining”. In: *Intelligent Data Engineering and Automated Learning: Proceedings of the 8th International Conference on Intelligent Data Engineering and Automated Learning.* 8th International Conference on Intelligent Data Engineering and Automated Learning (IDEAL07), Birmingham, UK, ed. by H. Yin, P. Tino, E. Corchado, et al. IDEAL ’07. Lecture Notes in Computer Science, Vol. 4881. Springer, pp. 695–704. DOI: 10.1007/978-3-540-77226-2_70.

[67] Donato, D., F. Bonchi, T. Chi, and Y. Maarek (2010). “Do You Want To Take Notes? Identifying Research Missions in Yahoo! Search Pad”. In: *Proceedings of the 19th International Conference on World Wide Web.* 19th International Conference on World Wide Web (WWW10), Raleigh, NC, USA, ed. by M. Rappa, P. Jones, J. Freire, et al. WWW ’10. Association for Computing Machinery, pp. 321–330. DOI: 10.1145/1772690.1772724.

[68] Downey, D., S. Dumais, and E. Horvitz (2007). “Models of Searching and Browsing: Languages, Studies, and Application”. In: *Proceedings of the 20th International Joint

Conference on Artifical Intelligence. 20th International Joint Conference on Artifical Intelligence (IJCAI07), Hyderabad, India, ed. by R. Sangal, H. Mehta, and R. K. Bagga. IJCAI '07. Morgan Kaufman Publishers Inc., pp. 2740–2747.

[69] Du, C., P. Shu, and Y. Li (2018). "CA-LSTM: Search Task Identification with Context Attention Based LSTM". In: *Proceedings of the 41st International ACM SIGIR Conference on Research & Development in Information Retrieval.* 41st International ACM SIGIR Conference on Research & Development in Information Retrieval (SIGIR18), Ann Arbor, MI, USA, ed. by K. Collins-Thompson, Q. Mei, B. Davison, et al. SIGIR '18. Association for Computing Machinery, pp. 1101–1104. DOI: 10.1145/3209978.3210087.

[70] Du, J. T. and A. Spink (2011). "Toward a Web Search Model: Integrating Multitasking, Cognitive Coordination, and Cognitive Shifts". In: *Journal of the American Society for Information Science and Technology* 62.8, pp. 1446–1472. DOI: 10.1002/asi.21551.

[71] Ekstrand, M. D., J. T. Riedl, and J. A. Konstan (2011). "Collaborative Filtering Recommender Systems". In: *Foundations and Trends in Human–Computer Interaction* 4.2, pp. 81–173. DOI: 10.1561/1100000009.

[72] Elekes, A., M. Schaeler, and K. Boehm (2017). "On the Various Semantics of Similarity in Word Embedding Models". In: *Proceedings of the 17th ACM/IEEE Joint Conference on Digital Libraries.* 2017 ACM/IEEE Joint Conference on Digital Libraries (JCDL17), Toronto, ON, Canada. JCDL '17. IEEE, pp. 139–148. DOI: 10.1109/JCDL.2017.7991568.

[73] Epure, E. V., B. Kille, J. E. Ingvaldsen, R. Deneckere, C. Salinesi, and S. Albayrak (2017). "Recommending Personalized News in Short User Sessions". In: *Proceedings of the 11th ACM Conference on Recommender Systems.* 11th ACM Conference on Recommender Systems (RecSys17), Como, Italy, ed. by P. Cremonesi, F. Ricci, S. Berkovsky, et al. RecSys '17. Association for Computing Machinery, pp. 121–129. DOI: 10.1145/3109859.3109894.

[74] Ester, M., H.-P. Kriegel, J. Sander, and X. Xu (1996). "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise". In: *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining.* 2nd International Conference on Knowledge Discovery and Data Mining (KDD96), Portland, OR, USA, ed. by E. Simoudis, J. Han, and U. Fayyad. KDD '96. AAAI Press, pp. 226–231. DOI: 10.5555/3001460.3001507.

[75] Fatima, B., H. Ramzan, and S. Asghar (2016). "Session Identification Techniques Used in Web Usage Mining: A Systematic Mapping of Scholarly Literature". In: *Online Information Review* 40.7, pp. 1033–1053. DOI: 10.1108/OIR-08-2015-0274.

[76] Feild, H. and J. Allan (2013). "Task-Aware Query Recommendation". In: *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval.* 36th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR13), Dublin, Ireland, ed. by G. J. Jones, P. Sheridan, D. Kelly, et al. SIGIR '13. Association for Computing Machinery, pp. 83–92. DOI: 10.1145/2484028.2484069.

[77] Filali, K., A. Nair, and C. Leggetter (2010). "Transitive History-Based Query Disambiguation for Query Reformulation". In: *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval.* 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR10), Geneva, Switzerland, ed. by F. Crestani, S. Marchand-Maillet, H.-H. Chen, et al. SIGIR '10. Association for Computing Machinery, pp. 849–850. DOI: 10.1145/1835449.1835647.

[78] Fox, S., K. Karnawat, M. Mydland, S. Dumais, and T. White (2005). "Evaluating Implicit Measures to Improve Web Search". In: *ACM Transactions on Information Systems* 23.2, pp. 147–168. DOI: 10.1145/1059981.1059982.

[79] Gabrilovich, E. and S. Markovitch (2007). "Computing Semantic Relatedness Using Wikipedia-based Explicit Semantic Analysis". In: *Proceedings of the 20th International Joint Conference on Artifical Intelligence.* 20th International Joint Conference on Artifical Intelligence (IJCAI07), Hyderabad, India, ed. by R. Sangal, H. Mehta, and R. K. Bagga. IJCAI '07. Morgan Kaufmann Publishers Inc., pp. 1606–1611. DOI: 10.5555/1625275.1625535.

[80] Gayo-Avello, D. (2009). "A Survey on Session Detection Methods in Query Logs and a Proposal for Future Evaluation". In: *Information Sciences* 179.12, pp. 1822–1843. DOI: 10.1016/j.ins.2009.01.026.

[81] Ghosh, D. and A. Vogt (2002). "Sampling Methods Related to Bernoulli and Poisson Sampling". In: *Proceedings of the Joint Statistical Meetings: Survey Research Methods Section.* Joint Statistical Meetings (JSM02), New York City, NY, USA. JSM '02. American Statistical Association, pp. 3569–3570.

[82] Göker, A. and D. He (2000). "Analysing Web Search Logs to Determine Session Boundaries for User-Oriented Learning". In: *Adaptive Hypermedia and Adaptive Web-Based Systems: Proceedings of the International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems.* International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH00), Trento, Italy, ed. by P. Brusilovsky, O. Stock, and C. Strapparava. AH '00. Lecture Notes in Computer Science, Vol. 1892. Springer, pp. 319–322. DOI: 10.1007/3-540-44595-1_38.

[83] Gomes, P., B. Martins, and L. Cruz (2019). "Segmenting User Sessions in Search Engine Query Logs Leveraging Word Embeddings". In: *Digital Libraries for Open Knowledge: Proceedings of the International Conference on Theory and Practice of Digital Libraries.* International Conference on Theory and Practice of Digital Libraries (TPDL19), Oslo, Norway, ed. by A. Doucet, A. Isaac, K. Golub, et al. TPDL '19. Lecture Notes in Computer Science, Vol. 11799. Springer, pp. 185–199. DOI: 10.1007/978-3-030-30760-8_17.

[84] Grbovic, M., N. Djuric, V. Radosavljevic, F. Silvestri, R. Baeza-Yates, A. Feng, E. Ordentlich, L. Yang, and G. Owens (2016). "Scalable Semantic Matching of Queries to Ads in Sponsored Search Advertising". In: *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval.* 39th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR16), Pisa, Italy, ed. by R. Perego, F. Sebastiani, J. Aslam, et al. SIGIR

'16. Association for Computing Machinery, pp. 375–384. DOI: 10.1145/2911451.2911538..

[85] Grbovic, M., V. Radosavljevic, N. Djuric, N. Bhamidipati, J. Savla, V. Bhagwan, and D. Sharp (2015). "E-Commerce in Your Inbox: Product Recommendations at Scale". In: *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD15), Sydney, NSW, Australia, ed. by L. Cao, C. Zhang, T. Joachims, et al. KDD '15. Association for Computing Machinery, pp. 1809–1818. DOI: 10.1145/2783258.2788627.

[86] Gu, Y., Z. Ding, S. Wang, and D. Yin (2020). "Hierarchical User Profiling for E-commerce Recommender Systems". In: *Proceedings of the 13th International Conference on Web Search and Data Mining.* 13th International Conference on Web Search and Data Mining (WSDM20), Houston, TX, USA, ed. by J. Caverlee, X. B. Hu, M. Lalmas, et al. WSDM '20. Association for Computing Machinery, pp. 223–231. DOI: 10.1145/3336191.3371827.

[87] Guha, N. (2017). "Semantic Identification of Web Browsing Sessions". In: *CoRR.* Arxiv abs/1704.03138.

[88] Hagen, M., J. Gomoll, A. Beyer, and B. Stein (2013). "From Search Session Detection to Search Mission Detection". In: *Proceedings of the 10th Conference on Open Research Areas in Information Retrieval.* 10th Conference on Open Research Areas in Information Retrieval (OAIR13), Lisbon, Portugal, ed. by J. Ferreira, J. Magalhães, and P. Calado. OAIR '13. Le Centre De Hautes Etudes Internationales D'Informatique Documentaire, pp. 85–92. DOI: 10.5555/2491748.2491769.

[89] Hagen, M., B. Stein, and T. Rüb (2011). "Query Session Detection as a Cascade". In: *Proceedings of the 20th ACM International Conference on Information and Knowledge Management.* 20th ACM International Conference on Information and Knowledge Management (CIKM11), Glasgow, Scotland, UK, ed. by B. Berendt, A. de Vries, W. Fan, et al. CIKM '11. Association for Computing Machinery, pp. 147–152. DOI: 10.1145/2063576.2063602.

[90] Halder, K., H.-T. Cheng, E. K. I. Chio, G. Roumpos, T. Wu, and R. Agarwal (2020). "Modeling Information Need of Users in Search Sessions". In: *CoRR.* Arxiv abs/2001.00861.

[91] Halfaker, A., O. Keyes, D. Kluver, J. Thebault-Spieker, T. Nguyen, K. Shores, A. Uduwage, and M. Warncke-Wang (2015). "User Session Identification Based on Strong Regularities in Inter-activity Time". In: *Proceedings of the 24th International Conference on World Wide Web.* 24th International Conference on World Wide Web (WWW15), Florence, Italy, ed. by A. Gangemi, S. Leonardi, and A. Panconesi. WWW '15. International World Wide Web Conferences Steering Committee, pp. 410–418. DOI: 10.1145/2736277.2741117.

[92] Han, S., D. He, and Y. Chi (2017). "Understanding and Modeling Behavior Patterns in Cross-device Web Search". In: *Proceedings of the Association for Information Science and Technology* 54.1, pp. 150–158. DOI: 10.1002/pra2.2017.14505401017.

[93] Hassan, A., R. Jones, and K. L. Klinkner (2010). “Beyond DCG: User Behavior as a Predictor of a Successful Search”. In: *Proceedings of the 3rd ACM International Conference on Web Search and Data Mining*. 3rd ACM International Conference on Web Search and Data Mining (WSDM10), New York City, NY, USA, ed. by B. D. Davison, T. Suel, N. Craswell, et al. WSDM ’10. Association for Computing Machinery, pp. 221–230. DOI: 10.1145/1718487.1718515.

[94] Hassan, A. and R. W. White (2013). “Personalized Models of Search Satisfaction”. In: *Proceedings of the 22nd ACM International Conference on Information & Knowledge Management*. 22nd ACM International Conference on Information & Knowledge Management (CIKM13), San Francisco, CA, USA, ed. by Q. He, A. Iyengar, W. Nejdl, et al. CIKM ’13. Association for Computing Machinery, pp. 2009–2018. DOI: 10.1145/2505515.2505681.

[95] He, D. and A. Göker (2000). “Detecting Session Boundaries from Web User Logs”. In: *Proceedings of the BCS/IRSG 22nd Annual Colloquium on Information Retrieval Research*. 22nd Annual Colloquium on Information Retrieval Research (BCS/IRSG00), Cambridge, UK. BCS/IRSG ’00, pp. 57–66.

[96] He, D., A. Göker, and D. J. Harper (2002). “Combining Evidence for Automatic Web Session Identification”. In: *Information Processing & Management* 38.5, pp. 727–742. DOI: 10.1016/S0306-4573(01)00060-7.

[97] He, X. and Q. Wang (2011). “Dynamic Timeout-Based a Session Identification Algorithm”. In: *Proceedings of the 2011 International Conference on Electric Information and Control Engineering*. 2011 International Conference on Electric Information and Control Engineering (ICEICE11), Wuhan, China. ICEICE ’11. IEEE, pp. 346–349. DOI: 10.1109/ICEICE.2011.5777587.

[98] Heer, J. and E. H. Chi (2002). “Separating the Swarm: Categorization Methods for User Sessions on the Web”. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. SIGCHI Conference on Human Factors in Computing Systems (CHI02), Minneapolis, MN, USA, ed. by D. Wixon. CHI ’02. Association for Computing Machinery, pp. 243–250. DOI: 10.1145/503376.503420.

[99] Hennig, C. (2015). “What Are the True Clusters?” In: *Pattern Recognition Letters*. Philosophical Aspects of Pattern Recognition 64, pp. 53–62. DOI: 10.1016/j.patrec.2015.04.009.

[100] Heydari, M., R. A. Helal, and K. I. Ghauth (2009). “A Graph-Based Web Usage Mining Method Considering Client Side Data”. In: *Proceedings of the 2009 International Conference on Electrical Engineering and Informatics*. 2009 International Conference on Electrical Engineering and Informatics (ICEEI09), Bangi, Malaysia. IEEE, pp. 147–153. DOI: 10.1109/ICEEI.2009.5254802.

[101] Hidasi, B. and A. Karatzoglou (2018). “Recurrent Neural Networks with Top-k Gains for Session-based Recommendations”. In: *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. 27th ACM International Conference on Information and Knowledge Management (CIKM18), Torino, Italy, ed. by A.

Cuzzocrea, J. Allan, N. Paton, et al. CIKM '18. Association for Computing Machinery, pp. 843–852. DOI: 10.1145/3269206.3271761.

[102] Hidasi, B., A. Karatzoglou, L. Baltrunas, and D. Tikk (2016). "Session-Based Recommendations with Recurrent Neural Networks". In: *Proceedings of the 4th International Conference on Learning Representations.* 4th International Conference on Learning Representations (ICLR16), San Juan, Puerto Rico, ed. by Y. Bengio and Y. LeCun. ICLR '16.

[103] Hienert, D. and D. Kern (2019). "Recognizing Topic Change in Search Sessions of Digital Libraries Based on Thesaurus and Classification System". In: *Proceedings of the 18th Joint Conference on Digital Libraries.* 18th Joint Conference on Digital Libraries (JCDL19), Champaign, IL, USA. JCDL '19. IEEE, pp. 297–300. DOI: 10.1109/JCDL.2019.00049.

[104] Hoxha, J., M. Junghans, and S. Agarwal (2012). "Enabling Semantic Analysis of User Browsing Patterns in the Web of Data". In: *Proceedings of the 2nd International Workshop on Usage Analysis and the Web of Data (USEWOD).* 2nd International Workshop on Usage Analysis and the Web of Data (USEWOD2012) , Co-Located with the 21st International World Wide Web Conference (WWW2012), Lyon, France. WWW '12. Arxiv.

[105] Hu, D., R. Louca, L. Hong, and J. McAuley (2018). "Learning Within-Session Budgets from Browsing Trajectories". In: *Proceedings of the 12th ACM Conference on Recommender Systems.* 12th ACM Conference on Recommender Systems (RecSys18), Vancouver, BC, Canada, ed. by S. Pera, M. Ekstrand, X. Amatriain, et al. RecSys '18. Association for Computing Machinery, pp. 432–436. DOI: 10.1145/3240323.3240401.

[106] Hua, W., Y. Song, H. Wang, and X. Zhou (2013). "Identifying Users' Topical Tasks in Web Search". In: *Proceedings of the 6th ACM International Conference on Web Search and Data Mining.* 6th ACM International Conference on Web Search and Data Mining (WSDM13), Rome, Italy, ed. by S. Leonardi, A. Panconesi, P. Ferragina, et al. WSDM '13. Association for Computing Machinery, pp. 93–102. DOI: 10.1145/2433396.2433410.

[107] Huang, C.-k., L.-f. Chien, and Y.-j. Oyang (2003). "Relevant Term Suggestion in Interactive Web Search Based on Contextual Information in Query Session Logs". In: *Journal of the American Society for Information Science and Technology* 54.7, pp. 638–649. DOI: 10.1002/asi.10256.

[108] Huang, X., F. Peng, A. An, and D. Schuurmans (2004). "Dynamic Web Log Session Identification with Statistical Language Models". In: *Journal of the American Society for Information Science and Technology* 55.14, pp. 1290–1303. DOI: 10.1002/asi.20084.

[109] Israel, G. D. (1992). *Determining Sample Size.* PEOD6. Gainesville, FL, USA: University Of Florida, Agricultural Education and Communication Department, Florida Cooperative Extension Service, Institute of Food and Agricultural Sciences.

[110] Janiszewski, C. (1998). “The Influence of Display Characteristics on Visual Exploratory Search Behavior”. In: *Journal of Consumer Research* 25.3, pp. 290–301. DOI: 10.1086/209540.

[111] Jansen, B. J. and A. Spink (2000). “Methodological Approach in Discovering User Search Patterns through Web Log Analysis”. In: *Bulletin of the American Society for Information Science and Technology* 27.1, pp. 15–17. DOI: 10.1002/bult.185.

[112] Jansen, B. J., A. Spink, J. Bateman, and T. Saracevic (1998). “Real Life Information Retrieval: A Study of User Queries on the Web”. In: *ACM SIGIR Forum* 32.1, pp. 5–17. DOI: 10.1145/281250.281253.

[113] Jansen, B. J., A. Spink, C. Blakely, and S. Koshman (2007). “Defining a Session on Web Search Engines”. In: *Journal of the American Society for Information Science and Technology* 58.6, pp. 862–871. DOI: 10.1002/asi.20564.

[114] Jansen, B. J., A. Spink, and B. Narayan (2007). “Query Modifications Patterns During Web Searching”. In: *Proceedings of the 4th International Conference on Information Technology New Generations*. 4th International Conference on Information Technology New Generations (ITNG07), Las Vegas, NV, USA. ITNG ’07. IEEE, pp. 439–444. DOI: 10.1109/ITNG.2007.164.

[115] Jansen, B. J., A. Spink, and J. Pedersen (2005). “A Temporal Comparison of AltaVista Web Searching”. In: *Journal of the American Society for Information Science and Technology* 56.6, pp. 559–570. DOI: 10.1002/asi.20145.

[116] Jiang, D., J. Pei, and H. Li (2013). “Mining Search and Browse Logs for Web Search: A Survey”. In: *ACM Transactions on Intelligent Systems and Technology* 4.4, pp. 1–37. DOI: 10.1145/2508037.2508038.

[117] Jiang, Y., Y. Li, C. Yang, E. M. Armstrong, T. Huang, and D. Moroni (2016). “Reconstructing Sessions from Data Discovery and Access Logs to Build a Semantic Knowledge Base for Improving Data Discovery”. In: *ISPRS International Journal of Geo-Information* 5.5. DOI: 10.3390/ijgi5050054.

[118] Joachims, T. (2002). “Optimizing Search Engines Using Clickthrough Data”. In: *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD02), Edmonton, AB, Canada, ed. by O. R. Zaïane, R. Goebel, D. Hand, et al. KDD ’02. Association for Computing Machinery, pp. 133–142. DOI: 10.1145/775047.775067.

[119] Joachims, T., L. Granka, B. Pan, H. Hembrooke, and G. Gay (2017). “Accurately Interpreting Clickthrough Data as Implicit Feedback”. In: *ACM SIGIR Forum* 51.1, pp. 4–11. DOI: 10.1145/3130332.3130334.

[120] Jones, R. and K. L. Klinkner (2008). “Beyond the Session Timeout: Automatic Hierarchical Segmentation of Search Topics in Query Logs”. In: *Proceedings of the 17th ACM Conference on Information and Knowledge Management*. 17th ACM Conference on Information and Knowledge Management (CIKM08), Napa Valley, CA, USA, ed.

by J. G. Shanahan, S. Amer-Yahia, I. Manolescu, et al. CIKM '08. Association for Computing Machinery, pp. 699–708. DOI: 10.1145/1458082.1458176.

[121] Jones, R., B. Rey, O. Madani, and W. Greiner (2006). "Generating Query Substitutions". In: *Proceedings of the 15th International Conference on World Wide Web.* 15th International Conference on World Wide Web (WWW06), Edinburgh, Scotland, UK, ed. by L. Carr, D. De Roure, A. Iyengar, et al. WWW '06. Association for Computing Machinery, pp. 387–396. DOI: 10.1145/1135777.1135835.

[122] Kamphuis, C., A. P. de Vries, L. Boytsov, and J. Lin (2020). "Which BM25 Do You Mean? A Large-Scale Reproducibility Study of Scoring Variants". In: *Advances in Information Retrieval, Part II: Proceedings of the 42nd European Conference on Information Retrieval Research.* 42nd European Conference on Information Retrieval Research (ECIR2020), Lisbon, Portugal, ed. by J. M. Jose, E. Yilmaz, J. Magalhães, et al. ECIR '20. Lecture Notes in Computer Science, Vol. 12036. Springer International Publishing, pp. 28–34. DOI: 10.1007/978-3-030-45442-5_4.

[123] Kannadasan, M. R. and G. Aslanyan (2019). "Personalized Query Auto-Completion Through a Lightweight Representation of the User Context". In: *Proceedings of the SIGIR 2019 Workshop on eCommerce, Co-Located with the 42st International ACM SIGIR Conference on Research and Development in Information Retrieval.* The SIGIR 2019 Workshop on eCommerce (eCOM19), Co-Located with the 42st International {ACM} {SIGIR} Conference on Research and Development in Information Retrieval (eCom@SIGIR19), Paris, France, ed. by J. Degenhardt, S. Kallumadi, U. Porwal, et al. eCom@SIGIR '19. CEUR Workshop Proceedings, Vol. 2410. CEUR-WS.org.

[124] Kanoulas, E., E. Yilmaz, R. Mehrotra, B. Carterette, N. Craswell, and P. Bailey (2017). "TREC 2017 Tasks Track Overview". In: *Proceedings of the 26th Text REtrieval Conference.* The 26th Text REtrieval Conference (TREC17), Gaithersburg, MD, USA, ed. by E. M. Voorhees and A. Ellis. TREC '17. Special Publication, Vol. 500-324. National Institute of Standards and Technology.

[125] Kapusta, J., M. Munk, P. Svec, and A. Pilkova (2014). "Determining the Time Window Threshold to Identify User Sessions of Stakeholders of a Commercial Bank Portal". In: *Procedia Computer Science* 29, pp. 1779–1790. DOI: 10.1016/j.procs.2014.05.163.

[126] Kaya, M. and D. Bridge (2019). "A Comparison of Calibrated and Intent-Aware Recommendations". In: *Proceedings of the 13th ACM Conference on Recommender Systems.* 13th ACM Conference on Recommender Systems (RecSys19), Copenhagen, Denmark, ed. by T. Bogers, A. Said, P. Brusilovsky, et al. RecSys '19. Association for Computing Machinery, pp. 151–159. DOI: 10.1145/3298689.3347045.

[127] Kelly, D. and N. J. Belkin (2004). "Display Time as Implicit Feedback: Understanding Task Effects". In: *Proceedings of the 27th International ACM SIGIR Conference on Research and Development in Information Retrieval.* 27th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR04), Sheffield, UK, ed. by M. Sanderson, K. Järvelin, J. Allan, et al. SIGIR '04. Association for Computing Machinery, pp. 377–384. DOI: 10.1145/1008992.1009057.

[128] Kharitonov, E., C. Macdonald, P. Serdyukov, and I. Ounis (2013). “Intent Models for Contextualising and Diversifying Query Suggestions”. In: *Proceedings of the 22nd ACM International Conference on Information & Knowledge Management.* 22nd ACM International Conference on Information & Knowledge Management (CIKM13), San Francisco, CA, USA, ed. by Q. He, A. Iyengar, W. Nejdl, et al. CIKM ’13. Association for Computing Machinery, pp. 2303–2308. DOI: 10.1145/2505515.2505661.

[129] Kim, Y., A. Hassan, R. W. White, and I. Zitouni (2014). “Modeling Dwell Time to Predict Click-level Satisfaction”. In: *Proceedings of the 7th ACM International Conference on Web Search and Data Mining.* 7th International ACM Conference on Web Search and Data Mining (WSDM2014), New York City, NY, USA, ed. by B. Carterette, F. Diaz, C. Castillo, et al. WSDM ’14. Association for Computing Machinery, pp. 193–202. DOI: 10.1145/2556195.2556220.

[130] Kiseleva, J., H. Thanh Lam, M. Pechenizkiy, and T. Calders (2013). “Discovering Temporal Hidden Contexts in Web Sessions for User Trail Prediction”. In: *Proceedings of the 22nd International Conference on World Wide Web.* 22nd International Conference on World Wide Web (WWW13), Rio de Janeiro, Brazil, ed. by D. Schwabe, V. Almeida, H. Glaser, et al. WWW ’13. Association for Computing Machinery, pp. 1067–1074. DOI: 10.1145/2487788.2488120.

[131] Kotov, A., P. N. Bennett, R. W. White, S. T. Dumais, and J. Teevan (2011). “Modeling and Analysis of Cross-Session Search Tasks”. In: *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information.* 34th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR11), Beijing, China, ed. by W.-Y. Ma, J.-Y. Nie, R. Baeza-Yates, et al. SIGIR ’11. Association for Computing Machinery, pp. 5–14. DOI: 10.1145/2009916.2009922.

[132] Kusner, M. J., Y. Sun, N. I. Kolkin, and K. Q. Weinberger (2015). “From Word Embeddings To Document Distances”. In: *Proceedings of the 32nd International Conference on Machine Learning - Volume 37.* 32nd International Conference on Machine Learning (ICML15), Lille, France, ed. by F. Bach and D. Blei. ICML ’15. JMLR.org, pp. 957–966. DOI: 10.5555/3045118.3045221.

[133] Lau, T. and E. Horvitz (1999). “Patterns of Search: Analyzing and Modeling Web Query Refinement”. In: *Proceedings of the 7th International Conference on User Modeling.* 7th International Conference on User Modeling (UM99), Vienna, Austria, ed. by J. Kay. USM ’99. International Centre for Mechanical Sciences, Vol. 407. Springer, pp. 119–128. DOI: 10.1007/978-3-7091-2490-1_12.

[134] Levine, N., H. Roitman, and D. Cohen (2017). “An Extended Relevance Model for Session Search”. In: *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval.* 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR17), Shinjuku, Tokyo, Japan, ed. by N. Kando, T. Sakai, H. Joho, et al. SIGIR ’17. Association for Computing Machinery, pp. 865–868. DOI: 10.1145/3077136.3080664.

[135] Levy, O. and Y. Goldberg (2014). “Dependency-Based Word Embeddings”. In: *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Short Papers).* 52nd Annual Meeting of the Association for Computational Linguistics

(ACL14), Baltimore, MD, USA, ed. by K. Toutanova and H. Wu. ACL '14. Association for Computational Linguistics, pp. 302–308. DOI: 10.3115/v1/P14-2050.

[136] Li, J., P. Ren, Z. Chen, Z. Ren, and J. Ma (2017). "Neural Attentive Session-based Recommendation". In: *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management.* 2017 ACM on Conference on Information and Knowledge Management (CIKM17), Singapore, Singapore, ed. by E.-P. Lim, M. Winslett, M. Sanderson, et al. CIKM '17. Association for Computing Machinery, pp. 1419–1428. DOI: 10.1145/3132847.3132926.

[137] Li, J., D. Song, P. Zhang, and Y. Hou (2015). "How Different Features Contribute to the Session Search?" In: *Natural Language Processing and Chinese Computing: Proceedings of the 4th CCF Conference on Natural Language Processing and Chinese Computing.* 4th CCF Conference on Natural Language Processing and Chinese Computing (NLPCC15), Nanchang, China, ed. by J. Li, H. Ji, D. Zhao, et al. NLPCC '15. Lecture Notes in Computer Science, Vol. 9362. Springer, pp. 242–253. DOI: 10.1007/978-3-319-25207-0_21.

[138] Li, L., H. Deng, A. Dong, Y. Chang, and H. Zha (2014). "Identifying and Labeling Search Tasks via Query-Based Hawkes Processes". In: *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD14), New York City, NY, USA, ed. by S. Macskassy, C. Perlich, J. Leskovec, et al. KDD '14. Association for Computing Machinery, pp. 731–740. DOI: 10.1145/2623330.2623679.

[139] Li, L., H. Deng, Y. He, A. Dong, Y. Chang, and H. Zha (2016). "Behavior Driven Topic Transition for Search Task Identification". In: *Proceedings of the 25th International Conference on World Wide Web.* 25th International Conference on World Wide Web (WWW16), Montréal, QC, Canada, ed. by J. Bourdeau, J. A. Hendler, R. Nkambou, et al. WWW '16. International World Wide Web Conferences Steering Committee, pp. 555–565. DOI: 10.1145/2872427.2883047.

[140] Li, X. (2018). "Mining Information Interaction Behavior: Academic Papers and Enterprise Emails". PhD thesis. Amsterdam, Netherlands: University of Amsterdam.

[141] Li, Y. (2020). "Interruption and Renewal: How and Why Do People Search Across Sessions?" In: *Proceedings of the 2020 Conference on Human Information Interaction and Retrieval.* 2020 Conference on Human Information Interaction and Retrieval (CHIIR20), Vancouver, BC, Canada, ed. by H. O'Brien, L. Freund, I. Arapakis, et al. CHIIR '20. Association for Computing Machinery, pp. 527–530. DOI: 10.1145/3343413.3377952.

[142] Li, Y., R. Capra, and Y. Zhang (2020). "Everyday Cross-session Search: How and Why Do People Search Across Multiple Sessions?" In: *Proceedings of the 2020 Conference on Human Information Interaction and Retrieval.* 2020 Conference on Human Information Interaction and Retrieval (CHIIR20), Vancouver, BC, Canada, ed. by H. O'Brien, L. Freund, I. Arapakis, et al. CHIIR '20. Association for Computing Machinery, pp. 163–172. DOI: 10.1145/3343413.3377970.

[143] Liao, Z., D. Jiang, J. Pei, Y. Huang, E. Chen, H. Cao, and H. Li (2013). “A vlHMM Approach to Context-Aware Search”. In: *ACM Transactions on the Web* 7.4, pp. 1–38. DOI: 10.1145/2490255.

[144] Liao, Z., Y. Song, L.-w. He, and Y. Huang (2012). “Evaluating the Effectiveness of Search Task Trails”. In: *Proceedings of the 21st International Conference on World Wide Web.* 21st International Conference on World Wide Web (WWW12), Lyon, France, ed. by A. Mille, F. Gandon, J. Misselis, et al. WWW ’12. Association for Computing Machinery, pp. 489–498. DOI: 10.1145/2187836.2187903.

[145] Liao, Z., Y. Song, Y. Huang, L.-w. He, and Q. He (2014). “Task Trail: An Effective Segmentation of User Search Behavior”. In: *IEEE Transactions on Knowledge and Data Engineering* 26.12, pp. 3090–3102. DOI: 10.1109/TKDE.2014.2316794.

[146] Lin, J. and W. J. Wilbur (2009). “Modeling Actions of PubMed Users with N-Gram Language Models”. In: *Information Retrieval* 12.4, pp. 487–503. DOI: 10.1007/s10791-008-9067-7.

[147] Lin, S. and I. Xie (2013). “Behavioral Changes in Transmuting Multisession Successive Searches over the Web”. In: *Journal of the American Society for Information Science and Technology* 64.6, pp. 1259–1283. DOI: 10.1002/asi.22839.

[148] Lindén, M. (2016). “Path Analysis of Online Users Using Clickstream Data: Case Online Magazine Website.” MA thesis. Lappeenranta, Finland: Lappeenranta University of Technology. 75 pp.

[149] Liu, C., R. W. White, and S. Dumais (2010). “Understanding Web Browsing Behaviors through Weibull Analysis of Dwell Time”. In: *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval.* 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR10), Geneva, Switzerland, ed. by F. Crestani, S. Marchand-Maillet, H.-H. Chen, et al. SIGIR ’10. Association for Computing Machinery, pp. 379–386. DOI: 10.1145/1835449.1835513.

[150] Liu, Y., C. Wang, M. Zhang, and S. Ma (2017). “User Behavior Modeling for Better Web Search Ranking”. In: *Frontiers of Computer Science* 11.6, pp. 923–936. DOI: 10.1007/s11704-017-6518-6.

[151] Liu, Z. and A. Zhang (2020). “A Survey on Sampling and Profiling over Big Data (Technical Report)”. In: *CoRR.* Arxiv abs/2005.05079.

[152] Loyola, P., C. Liu, and Y. Hirate (2017). “Modeling User Session and Intent with an Attention-based Encoder-Decoder Architecture”. In: *Proceedings of the 11th ACM Conference on Recommender Systems.* 11th ACM Conference on Recommender Systems (RecSys17), Como, Italy, ed. by P. Cremonesi, F. Ricci, S. Berkovsky, et al. RecSys ’17. Association for Computing Machinery, pp. 147–151. DOI: 10.1145/3109859.3109917.

[153] Lu, Z., H. Zha, X. Yang, W. Lin, and Z. Zheng (2013). “A New Algorithm for Inferring User Search Goals with Feedback Sessions”. In: *IEEE Transactions on Knowledge and Data Engineering* 25.3, pp. 502–513. DOI: 10.1109/TKDE.2011.248.

[154] Lucchese, C., S. Orlando, R. Perego, F. Silvestri, and G. Tolomei (2011). "Identifying Task-Based Sessions in Search Engine Query Logs". In: *Proceedings of the 4th ACM International Conference on Web Search and Data Mining.* 4th ACM International Conference on Web Search and Data Mining (WSDM11), Hong Kong, Hong Kong, ed. by I. King, W. Nejdl, and H. Li. WSDM '11. Association for Computing Machinery, pp. 277–286. DOI: 10.1145/1935826.1935875.

[155] — (2013). "Discovering Tasks from Search Engine Query Logs". In: *ACM Transactions on Information Systems* 31.3, pp. 1–43. DOI: 10.1145/2493175.2493179.

[156] Ludewig, M. and D. Jannach (2018). "Evaluation of Session-based Recommendation Algorithms". In: *User Modeling and User-Adapted Interaction* 28.4-5, pp. 331–390. DOI: 10.1007/s11257-018-9209-6.

[157] Lugo, L., J. G. Moreno, and G. Hubert (2020). "Segmenting Search Query Logs by Learning to Detect Search Task Boundaries". In: *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval.* 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR20), Virtual Event, ed. by J. Huang, Y. Chang, X. Cheng, et al. SIGIR '20. Association for Computing Machinery, pp. 2037–2040. DOI: 10.1145/3397271.3401257.

[158] — (2021). "Extracting Search Tasks from Query Logs Using a Recurrent Deep Clustering Architecture". In: *Advances in Information Retrieval: Proceedings of the 43rd European Conference on Information Retrieval Research.* 43rd European Conference on Information Retrieval Research (ECIR21), Virtual Event, ed. by D. Hiemstra, M.-F. Moens, J. Mothe, et al. ECIR '21. Lecture Notes in Computer Science, Vol. 12656. Springer, pp. 391–404. DOI: 10.1007/978-3-030-72113-8_26.

[159] Luxenburger, J., S. Elbassuoni, and G. Weikum (2008). "Matching Task Profiles and User Needs in Personalized Web Search". In: *Proceedings of the 17th ACM Conference on Information and Knowledge Management.* 17th ACM Conference on Information and Knowledge Management (CIKM08), Napa Valley, CA, USA, ed. by J. G. Shanahan, S. Amer-Yahia, I. Manolescu, et al. CIKM '08. Association for Computing Machinery, pp. 689–698. DOI: 10.1145/1458082.1458175.

[160] Lv, Y., L. Zhuang, and P. Luo (2019). "Neighborhood-Enhanced and Time-Aware Model for Session-based Recommendation". In: *CoRR.* Arxiv abs/1909.11252.

[161] Lv, Y. and C. Zhai (2011). "When Documents Are Very Long, BM25 Fails!" In: *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR11).* 34th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR11), Beijing, China, ed. by W.-Y. Ma, J.-Y. Nie, R. Baeza-Yates, et al. SIGIR '11. Association for Computing Machinery, pp. 1103–1104. DOI: 10.1145/2009916.2010070.

[162] MacKay, B. and C. Watters (2006). "Exploring Multiple Browser Session Behavior". In: *Proceedings of the American Society for Information Science and Technology* 43.1, pp. 1–4. DOI: 10.1002/meet.14504301279.

[163] MacKay, B. and C. Watters (2008a). “Exploring Multi-session Web Tasks”. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems.* SIGCHI Conference on Human Factors in Computing Systems (CHI08), Florence, Italy, ed. by M. Czerwinski, A. Lund, and D. Tan. CHI ’08. Association for Computing Machinery, pp. 1187–1196. DOI: 10.1145/1357054.1357243.

[164] — (2008b). “Understanding and Supporting Multi-Session Web Tasks”. In: *Proceedings of the American Society for Information Science and Technology* 45.1, pp. 1–13. DOI: 10.1002/meet.2008.1450450266.

[165] — (2012). “An Examination of Multisession Web Tasks”. In: *Journal of the American Society for Information Science and Technology* 63.6, pp. 1183–1197. DOI: 10.1002/asi.22610.

[166] Maguitman, A. (2018). “Searching in the Context of a Task: A Review of Methods and Tools”. In: *CLEI Electronic Journal* 21, p. 1. DOI: 10.19153/cleiej.21.1.1.

[167] Makhoul, J., F. Kubala, R. Schwartz, and R. Weischedel (1999). “Performance Measures For Information Extraction”. In: *Proceedings of DARPA Broadcast News Workshop.* Broadcast News Workshop (DARPA99), Herndon, VA, USA. DARPA ’99. Morgan Kaufmann Publishers Inc., pp. 249–252.

[168] Mandel, N. and E. J. Johnson (2002). “When Web Pages Influence Choice: Effects of Visual Primes on Experts and Novices”. In: *Journal of Consumer Research* 29.2, pp. 235–245. DOI: 10.1086/341573.

[169] Mandl, T. and D. Wilczek (2009). “Methoden für Robustes Information Retrieval und dessen Evaluierung”. In: *LWA 2009: Lernen, Wissen, Adaptivität.* Workshop-Woche: Lernen, Wissen, Adaptivität (LWA09), Darmstadt, Germany, ed. by M. Hartmann and F. Janssen. LWA ’09. FG Telekooperation/FG Knowledge Engineering, Technische Universität Darmstadt, pp. 72–75.

[170] Manning, C., P. Raghavan, and H. Schuetze (2008). *Introduction to Information Retrieval.* Cambridge University Press.

[171] Mayr, P. and A. Kacem (2017). “A Complete Year of User Retrieval Sessions in a Social Sciences Academic Search Engine”. In: *Research and Advanced Technology for Digital Libraries: Proceedings of the International Conference on Theory and Practice of Digital Libraries.* International Conference on Theory and Practice of Digital Libraries (TPDL17), Thessaloniki, Greece, ed. by J. Kamps, G. Tsakonas, Y. Manolopoulos, et al. TPDL ’17. Lecture Notes in Computer Science, Vol. 10450. Springer, pp. 560–565. DOI: 10.1007/978-3-319-67008-9_46.

[172] McInnes, L. and J. Healy (2017). “Accelerated Hierarchical Density Clustering”. In: *Proceedings of the 2017 IEEE International Conference on Data Mining.* 2017 IEEE International Conference on Data Mining Workshops (ICDMW), New Orleans, LA, USA, ed. by R. Gottumukkala, X. Ning, G. Dong, et al. ICDMW ’17. IEEE, pp. 33–42. DOI: 10.1109/ICDMW.2017.12.

[173] Mehrotra, R. (2019). “Inferring User Needs & Tasks from User Interactions”. In: *ACM SIGIR Forum* 52.2, pp. 176–177. DOI: 10.1145/3308774.3308806.

[174] Mehrotra, R., A. H. Awadallah, M. Shokouhi, E. Yilmaz, I. Zitouni, A. El Kholy, and M. Khabsa (2017a). “Deep Sequential Models for Task Satisfaction Prediction”. In: *Proceedings of the 2017 ACM Conference on Information and Knowledge Management.* 2017 ACM Conference on Information and Knowledge Management (CIKM17), Singapore, Singapore, ed. by E.-P. Lim, M. Winslett, M. Sanderson, et al. CIKM ’17. Association for Computing Machinery, pp. 737–746. DOI: 10.1145/3132847.3133001.

[175] Mehrotra, R., P. Bhattacharya, and E. Yilmaz (2016a). “Characterizing Users’ Multi-Tasking Behavior in Web Search”. In: *Proceedings of the 2016 ACM Conference on Human Information Interaction and Retrieval.* 2016 ACM Conference on Human Information Interaction and Retrieval (CHIIR16), Carrboro, NC, USA, ed. by D. Kelly, R. Capra, N. Belkin, et al. CHIIR ’16. Association for Computing Machinery, pp. 297–300. DOI: 10.1145/2854946.2855006.

[176] — (2016b). “Deconstructing Complex Search Tasks: A Bayesian Nonparametric Approach for Extracting Sub-tasks”. In: *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies.* 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL16), San Diego, CA, USA. NAACL ’16. Association for Computational Linguistics, pp. 599–605. DOI: 10.18653/v1/N16-1073.

[177] — (2016c). “Uncovering Task Based Behavioral Heterogeneities in Online Search Behavior”. In: *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval.* 39th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR16), Pisa, Italy, ed. by R. Perego, F. Sebastiani, J. Aslam, et al. SIGIR ’16. Association for Computing Machinery, pp. 1049–1052. DOI: 10.1145/2911451.2914755.

[178] Mehrotra, R., A. E. Kholy, I. Zitouni, M. Shokouhi, and A. Hassan (2017b). “Identifying User Sessions in Interactions with Intelligent Digital Assistants”. In: *Proceedings of the 26th International Conference on World Wide Web.* 26th International Conference on World Wide Web (WWW17), Perth, WA, Australia, ed. by R. Barrett, R. Cummings, E. Agichtein, et al. WWW ’17. International World Wide Web Conferences Steering Committee, pp. 821–822. DOI: 10.1145/3041021.3054254.

[179] Mehrotra, R. and E. Yilmaz (2015). “Towards Hierarchies of Search Tasks & Subtasks”. In: *Proceedings of the 24th International Conference on World Wide Web.* 24th International Conference on World Wide Web (WWW15), Florence, Italy, ed. by A. Gangemi, S. Leonardi, and A. Panconesi. WWW ’15. Association for Computing Machinery, pp. 73–74. DOI: 10.1145/2740908.2742777.

[180] — (2017a). “Extracting Hierarchies of Search Tasks & Subtasks via a Bayesian Nonparametric Approach”. In: *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval.* 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR17), Shinjuku, Tokyo, Japan, ed. by N. Kando, T. Sakai, H. Joho, et al. SIGIR ’17. Association for Computing Machinery, pp. 285–294. DOI: 10.1145/3077136.3080823.

[181] Mehrotra, R. and E. Yilmaz (2017b). “Task Embeddings: Learning Query Embeddings Using Task Context”. In: *Proceedings of the 2017 ACM Conference on Information and Knowledge Management.* 2017 ACM on Conference Information and Knowledge Management (CIKM17), Singapore, Singapore, ed. by E.-P. Lim, M. Winslett, M. Sanderson, et al. CIKM ’17. Association for Computing Machinery, pp. 2199–2202. DOI: 10.1145/3132847.3133098.

[182] Mehrzadi, D. and D. G. Feitelson (2012). “On Extracting Session Data from Activity Logs”. In: *Proceedings of the 5th International Systems and Storage Conference.* 5th International Systems and Storage Conference (SYSTOR12), Haifa, Israel, ed. by M. Vinov, D. Tsafrir, and E. Zadok. SYSTOR ’12. Association for Computing Machinery, pp. 1–7. DOI: 10.1145/2367589.2367592.

[183] Mei, Q., K. Klinkner, R. Kumar, and A. Tomkins (2009). “An Analysis Framework for Search Sequences”. In: *Proceedings of the 18th ACM Conference on Information and Knowledge Management.* 18th ACM Conference on Information and Knowledge Management (CIKM09), Hong Kong, Hong Kong, ed. by D. Cheung, I.-Y. Song, W. Chu, et al. CIKM ’09. Association for Computing Machinery, pp. 1991–1994. DOI: 10.1145/1645953.1646284.

[184] Meiss, M., J. Duncan, B. Gonçalves, J. J. Ramasco, and F. Menczer (2009). “What’s in a Session: Tracking Individual Behavior on the Web”. In: *Proceedings of the 20th ACM Conference on Hypertext and Hypermedia.* 20th ACM Conference on Hypertext and Hypermedia (HT09), Torino, Italy, ed. by C. Cattuto, G. Ruffo, and F. Menczer. HT ’09. Association for Computing Machinery, pp. 173–182. DOI: 10.1145/1557914.1557946.

[185] Menasalvas, E., S. Millán, J. M. Peña, M. Hadjimichael, and O. Marbán (2004). “Subsessions: A Granular Approach to Click Path Analysis”. In: *International Journal of Intelligent Systems* 19.7, pp. 619–637. DOI: 10.1002/int.20014.

[186] Mikolov, T., K. Chen, G. Corrado, and J. Dean (2013). “Efficient Estimation of Word Representations in Vector Space”. In: *International Conference on Learning Representations: Workshops Track.* International Conference on Learning Representations (ICLR13), Scottsdale, AZ, USA, ed. by Y. Bengio and Y. Lecun. ICLR ’13.

[187] Moe, W. W. (2003). “Buying, Searching, or Browsing: Differentiating Between Online Shoppers Using In-Store Navigational Clickstream”. In: *Journal of Consumer Psychology* 13.1-2, pp. 29–39. DOI: 10.1207/S15327663JCP13-1&2_03.

[188] Montgomery, A. and C. Faloutsos (2001). “Identifying Web Browsing Trends and Patterns”. In: *Computer* 34.7, pp. 94–95. DOI: 10.1109/2.933515.

[189] Montgomery, A. L., S. Li, K. Srinivasan, and J. C. Liechty (2004). “Modeling Online Browsing and Path Analysis Using Clickstream Data”. In: *Marketing Science* 23.4, pp. 579–595. DOI: 10.1287/mksc.1040.0073.

[190] Moreira, G. d. S. P. (2019). “CHAMELEON: A Deep Learning Meta-Architecture for News Recommender Systems”. PhD thesis. Sao Jose dos Campos, Brazil: Instituto Tecnologico de Aeronautica. 186 pp.

[191] Mothe, J. and L. Tanguy (2005). “Linguistic Features to Predict Query Difficulty - a Case Study on Previous TREC Campaigns”. In: ACM Conference on Research and Development in Information Retrieval (SIGIR05), Predicting Query Difficulty - Methods and Applications Workshop, Salvador de Bahia, Brazil. SIGIR ’05, pp. 7–10.

[192] Munk, M. and M. Drlík (2011). “Impact of Different Pre-Processing Tasks on Effective Identification of Users’ Behavioral Patterns in Web-based Educational System”. In: *Procedia Computer Science* 4, pp. 1640–1649. DOI: 10.1016/j.procs.2011.04.177.

[193] Murray, G. C., J. Lin, and A. Chowdhury (2007). “Identification of User Sessions with Hierarchical Agglomerative Clustering”. In: *Proceedings of the American Society for Information Science and Technology* 43.1, pp. 1–9. DOI: 10.1002/meet.14504301312.

[194] Nigam, P. and R. K. (2016). “A Comparative Analysis of Web Usage Mining Techniques”. In: *International Journal of Computer Applications* 152.5, pp. 26–29. DOI: 10.5120/ijca2016911790.

[195] Nikiforakis, N., A. Kapravelos, W. Joosen, C. Kruegel, F. Piessens, and G. Vigna (2013). “Cookieless Monster: Exploring the Ecosystem of Web-Based Device Fingerprinting”. In: *Proceedings of the 2013 IEEE Symposium on Security and Privacy.* 2013 IEEE Symposium on Security and Privacy (SP13), San Francisco, CA, USA. SP ’13. IEEE, pp. 541–555. DOI: 10.1109/SP.2013.43.

[196] Nottorf, F. (2013). “Which Clicks Lead to Conversions? - Modeling User-journeys Across Multiple Types of Online Advertising:” in: *Proceedings of the 4th International Conference on Data Communication Networking, 10th International Conference on e-Business and 4th International Conference on Optical Communication Systems.* 4th International Conference on Data Communication Networking, 10th International Conference on e-Business and 4th International Conference on Optical Communication Systems (ICETE2013), Reykjavík, Iceland, ed. by M. S. Obaidat, J. L. Sevillano, Z. Zhang, et al. Vol. 1. ICE-B ’13. SciTePress, pp. 141–152. DOI: 10.5220/0004504901410152.

[197] Obaid, H. S., S. A. Dheyab, and S. S. Sabry (2019). “The Impact of Data Pre-Processing Techniques and Dimensionality Reduction on the Accuracy of Machine Learning”. In: *Proceedings of the 2019 9th Annual Information Technology, Electromechanical Engineering and Microelectronics Conference (IEMECON).* 2019 9th Annual Information Technology, Electromechanical Engineering and Microelectronics Conference (IEMECON), Jaipur, India. IEMECON ’19. IEEE, pp. 279–283. DOI: 10.1109/IEMECONX.2019.8877011.

[198] Ortega, J. L. and I. Aguillo (2010). “Differences between Web Sessions According to the Origin of Their Visits”. In: *Journal of Informetrics* 4.3, pp. 331–337. DOI: 10.1016/j.joi.2010.02.001.

[199] Ozmutlu, H. C. and F. Çavdur (2005). “Application of Automatic Topic Identification on Excite Web Search Engine Data Logs”. In: *Information Processing & Management* 41.5, pp. 1243–1262. DOI: 10.1016/j.ipm.2004.04.018.

[200] Padala, V. K., S. Yasin, and D. B. Alanka (2013). “A Novel Method for Data Cleaning and User-Session Identification for Web Mining”. In: *International Journal of Modern Engineering Research* 3.5, pp. 2816–2819.

[201] Pallis, G., L. Angelis, and A. Vakali (2007). “Validation and Interpretation of Web Users’ Sessions Clusters”. In: *Information Processing & Management: An International Journal* 43.5, pp. 1348–1367. DOI: 10.1016/j.ipm.2006.10.010.

[202] Parvatikar, S. and B. Joshi (2014). “Analysis of User Behavior through Web Usage Mining”. In: *Proceedings on International Conference on Advances in Science and Technology.* International Conference on Advances in Science and Technology (IJCAST14). IJCAST ’14. IJCA Journal, pp. 27–31.

[203] Patel, K. and P. Bhattacharyya (2017). “Towards Lower Bounds on Number of Dimensions for Word Embeddings”. In: *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers).* Eighth International Joint Conference on Natural Language Processing (IJCNLP17), Taipei, Taiwan, ed. by G. Kondrak and T. Watanabe. Vol. 2. IJCNLP ’17. Asian Federation of Natural Language Processing, pp. 31–36.

[204] Patil, N. V. and H. D. Patil (2017). “Prediction of Web User’s Browsing Behavior Using All Kth Markov Model and CSB-mine”. In: *International Journal of Computer Trends and Technology* 43.1, pp. 68–74. DOI: 10.14445/22312803/IJCTT-V43P110.

[205] Patterson, J. and A. Gibson (2017). *Deep Learning - A Practitioner’s Approach.* O’Reilly Media, Inc.

[206] Pazzani, M. J. and D. Billsus (2007). “Content-Based Recommendation Systems”. In: *The Adaptive Web: Methods and Strategies of Web Personalization*, ed. by P. Brusilovsky, A. Kobsa, and W. Nejdl. Lecture Notes in Computer Science, Vol. 4321. Springer, pp. 325–341. DOI: 10.1007/978-3-540-72079-9_10.

[207] Peng, Z. and M.-s. Zhao (2010). “Session Identification Algorithm for Web Log Mining”. In: *Proceedings of the 2010 International Conference on Management and Service Science.* 2010 International Conference on Management and Service Science (MASS10), Wuhan, China. MASS ’10. IEEE, pp. 1–4. DOI: 10.1109/ICMSS.2010.5576547.

[208] Pi, Q., W. Bian, G. Zhou, X. Zhu, and K. Gai (2019). “Practice on Long Sequential User Behavior Modeling for Click-Through Rate Prediction”. In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining.* 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD19), Anchorage, AK, USA, ed. by A. Teredesai, V. Kumar, Y. Li, et al. KDD ’19. Association for Computing Machinery, pp. 2671–2679. DOI: 10.1145/3292500.3330666.

[209] Piwowarski, B., G. Dupret, and R. Jones (2009). “Mining User Web Search Activity with Layered Bayesian Networks or How to Capture a Click in Its Context”. In: *Proceedings of the 2nd ACM International Conference on Web Search and Data Mining.* 2nd ACM International Conference on Web Search and Data Mining (WSDM09), Barcelona, Spain, ed. by R. Baeza-Yates, P. Boldi, B. Ribeiro-Neto, et al. WSDM ’09. Association for Computing Machinery, pp. 162–171. DOI: 10.1145/1498759.1498823.

[210] Piwowarski, B. and H. Zaragoza (2007). “Predictive User Click Models Based on Click-through History”. In: *Proceedings of the 16th ACM Conference on Information and Knowledge Management.* 16th ACM Conference on Information and Knowledge

Management (CIKM07), Lisbon, Portugal, ed. by A. H. F. Laender, A. O. Falcão, Ø. H. Olsen, et al. CIKM '07. Association for Computing Machinery, pp. 175–182. DOI: 10.1145/1321440.1321467.

[211] Pratap, M., S. Dohare, P. Arya, and A. Bajpai (2012). "Novel Web Usage Mining for Web Mining Techniques". In: *International Journal of Emerging Technology and Advanced Engineering* 2.1, pp. 253–262. DOI: 10.1.1.414.529.

[212] Quadrana, M., P. Cremonesi, and D. Jannach (2019). "Sequence-Aware Recommender Systems". In: *ACM Computing Surveys* 51.4, pp. 1–36. DOI: 10.1145/3190616.

[213] Quadrana, M., A. Karatzoglou, B. Hidasi, and P. Cremonesi (2017). "Personalizing Session-based Recommendations with Hierarchical Recurrent Neural Networks". In: *Proceedings of the 11th ACM Conference on Recommender Systems.* 11th ACM Conference on Recommender Systems (RecSys17), Como, Italy, ed. by P. Cremonesi, F. Ricci, S. Berkovsky, et al. RecSys '17. Association for Computing Machinery, pp. 130–137. DOI: 10.1145/3109859.3109896.

[214] Rac, M. (2019). "User's Activity Driven Short-Term Context Inference". In: *Proceedings of the 13th ACM Conference on Recommender Systems.* 13th ACM Conference on Recommender Systems (RecSys19), Copenhagen, Denmark, ed. by T. Bogers, A. Said, P. Brusilovsky, et al. RecSys '19. Association for Computing Machinery, pp. 591–595. DOI: 10.1145/3298689.3346950.

[215] Radlinski, F. and T. Joachims (2005). "Query Chains: Learning to Rank from Implicit Feedback". In: *Proceedings of the 11th ACM SIGKDD International Conference on Knowledge Discovery in Data Mining.* 11th ACM SIGKDD International Conference on Knowledge Discovery in Data Mining (KDD05), Chicago, IL, USA, ed. by R. Grossman, R. Bayardo, and K. Bennett. KDD '05. Association for Computing Machinery, pp. 239–248. DOI: 10.1145/1081870.1081899.

[216] Radlinski, F., M. Szummer, and N. Craswell (2010). "Inferring Query Intent from Reformulations and Clicks". In: *Proceedings of the 19th International Conference on World Wide Web.* 19th International Conference on World Wide Web (WWW10), Raleigh, NC, USA, ed. by M. Rappa, P. Jones, J. Freire, et al. WWW '10. ACM Press, pp. 1171–1172. DOI: 10.1145/1772690.1772859.

[217] Raman, K., P. N. Bennett, and K. Collins-Thompson (2013). "Toward Whole-Session Relevance: Exploring Intrinsic Diversity in Web Search". In: *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval.* 36th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR13), Dublin, Ireland, ed. by G. J. F. Jones, P. Sheridan, D. Kelly, et al. SIGIR '13. Association for Computing Machinery, pp. 463–472. DOI: 10.1145/2484028.2484089.

[218] — (2014). "Understanding Intrinsic Diversity in Web Search: Improving Whole-Session Relevance". In: *ACM Transactions on Information Systems* 32.4, pp. 1–45. DOI: 10.1145/2629553.

[219] Řehůřek, R. and P. Sojka (2010). "Software Framework for Topic Modelling with Large Corpora". In: *Proceedings of the LREC 2010 Workshop on New Challenges for NLP.*

LREC 2010 Workshop on New Challenges for NLP Frameworks (LREC10), Valletta, Malta. LREC '10, pp. 45–50. DOI: 10.13140/2.1.2393.1847.

[220] Ren, K., J. Qin, Y. Fang, W. Zhang, L. Zheng, W. Bian, G. Zhou, J. Xu, Y. Yu, X. Zhu, and K. Gai (2019). "Lifelong Sequential Modeling with Personalized Memorization for User Response Prediction". In: *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval.* 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR19), Paris, France, ed. by B. Piwowarski, M. Chevalier, E. Gaussier, et al. SIGIR '19. Association for Computing Machinery, pp. 565–574. DOI: 10.1145/3331184.3331230.

[221] Reubold, J., A. Boubekki, T. Strufe, and U. Brefeld (2018). "Infinite Mixtures of Markov Chains". In: *New Frontiers in Mining Complex Patterns: Proceedings of the 6th International Workshop on New Frontiers in Mining Complex Patterns.* International Workshop on New Frontiers in Mining Complex Patterns (NFMCP17), Skopje, Macedonia, ed. by A. Appice, C. Loglisci, G. Manco, et al. NFMCP '17. Lecture Notes in Computer Science, Vol. 10785. Springer, pp. 167–181. DOI: 10.1007/978-3-319-78680-3_12.

[222] Richardson, M. (2008). "Learning about the World through Long-Term Query Logs". In: *ACM Transactions on the Web* 2.4, pp. 1–27. DOI: 10.1145/1409220.1409224.

[223] Robertson, S., S. Walker, S. Jones, M. Hancock-Beaulieu, and M. Gatford (1994). "Okapi at TREC-3". In: *Proceedings of the 3rd Text Retrieval Conference (TREC-3).* Third Text REtrieval Conference (TREC3), Gaithersburg, MD, USA, ed. by D. K. Harman. TREC-3. NIST Special Publication, Vol. 500-225. National Institute of Standards and Technology, pp. 109–126.

[224] Rohm, A. and V. Swaminathan (2004). "A Typology of Online Shoppers Based on Shopping Motivations". In: *Journal of Business Research* 57.7, pp. 748–757. DOI: 10.1016/S0148-2963(02)00351-X.

[225] Ruocco, M., O. S. L. Skrede, and H. Langseth (2017). "Inter-Session Modeling for Session-Based Recommendation". In: *Proceedings of the 2nd Workshop on Deep Learning for Recommender Systems.* 2nd Workshop on Deep Learning for Recommender Systems (DLRS17), Como, Italy, ed. by B. Hidasu, A. Karatzoglou, O. Sar-Shalom, et al. DLRS '17. Association for Computing Machinery, pp. 24–31. DOI: 10.1145/3125486.3125491.

[226] Sadikov, E., J. Madhavan, L. Wang, and A. Halevy (2010). "Clustering Query Refinements by User Intent". In: *Proceedings of the 19th International Conference on World Wide Web.* 19th International Conference on World Wide Web (WWW10), Raleigh, NC, USA, ed. by M. Rappa, P. Jones, J. Freire, et al. WWW '10. Association for Computing Machinery, pp. 841–850. DOI: 10.1145/1772690.1772776.

[227] Sánchez-Franco, M. J. and J. Rodríguez-Bobada Rey (2004). "Personal Factors Affecting Users' Web Session Lengths". In: *Internet Research* 14.1, pp. 62–80. DOI: 10.1108/10662240410516327.

[228] Sarwar, B., G. Karypis, J. Konstan, and J. Riedl (2001). “Item-Based Collaborative Filtering Recommendation Algorithms”. In: *Proceedings of the 10th International Conference on World Wide Web*. 10th International Conference on World Wide Web (WWW01), Hong Kong, Hong Kong, ed. by V. Y. Shen, N. Saito, M. R. Lyu, et al. WWW ’01. Association for Computing Machinery, pp. 285–295. DOI: 10.1145/371920.372071.

[229] Sen, P., D. Ganguly, and G. J. Jones (2018). “Tempo-Lexical Context Driven Word Embedding for Cross-Session Search Task Extraction”. In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. 2018 Conference of the North American Chapter of the Association for Computational Linguistics, New Orleans, LA, USA, ed. by M. Walker, H. Ji, and A. Stent. NAACL-HLT ’18. Association for Computational Linguistics, pp. 283–292. DOI: 10.18653/v1/N18-1026.

[230] Shen, X., B. Tan, and C. Zhai (2005). “Implicit User Modeling for Personalized Search”. In: *Proceedings of the 14th ACM International Conference on Information and Knowledge Management*. 14th ACM International Conference on Information and Knowledge Management (CIKM05), Bremen, Germany, ed. by O. Herzog, H.-J. Schek, N. Fuhr, et al. CIKM ’05. Association for Computing Machinery, pp. 824–831. DOI: 10.1145/1099554.1099747.

[231] Shi, X. and C. C. Yang (2006). “Mining Related Queries from Search Engine Query Logs”. In: *Proceedings of the 15th International Conference on World Wide Web*. 15th International Conference on World Wide Web (WWW06), Edinburgh, Scotland, UK, ed. by L. Carr, D. De Roure, A. Iyengar, et al. WWW ’06. Association for Computing Machinery, pp. 943–944. DOI: 10.1145/1135777.1135956.

[232] Silverstein, C., H. Marais, M. Henzinger, and M. Moricz (1999). “Analysis of a Very Large Web Search Engine Query Log”. In: *ACM SIGIR Forum* 33.1, pp. 6–12. DOI: 10.1145/331403.331405.

[233] Singh, L., S. Singh, S. Arora, and S. Borar (2019). “One Embedding To Do Them All”. In: *CoRR*. Arxiv abs/1906.12120.

[234] Sisodia, D. S., V. Khandal, and R. Singhal (2018). “Fast Prediction of Web User Browsing Behaviours Using Most Interesting Patterns”. In: *Journal of Information Science* 44.1, pp. 74–90. DOI: 10.1177/0165551516673293.

[235] Smirnova, E. and F. Vasile (2017). “Contextual Sequence Modeling for Recommendation with Recurrent Neural Networks”. In: *Proceedings of the 2nd Workshop on Deep Learning for Recommender Systems*. 2nd Workshop on Deep Learning for Recommender Systems (DLRS17), Como, Italy, ed. by B. Hidasi, A. Karatzoglou, O. Sar-Shalom, et al. DLRS ’17. Association for Computing Machinery, pp. 2–9. DOI: 10.1145/3125486.3125488.

[236] Song, Y., N. Sahoo, and E. Ofek (2019). “When and How to Diversify - A Multicategory Utility Model for Personalized Content Recommendation”. In: *Management Science* 65.8, pp. 3737–3757. DOI: 10.1287/mnsc.2018.3127.

[237] Sontag, D., K. Collins-Thompson, P. N. Bennett, R. W. White, S. Dumais, and B. Billerbeck (2012). “Probabilistic Models for Personalizing Web Search”. In: *Proceedings of the 5th ACM International Conference on Web Search and Data Mining.* 5th ACM International Conference on Web Search and Data Mining (WSDM12), Seattle, WA, USA, ed. by E. Adar, J. Teevan, E. Agichtein, et al. WSDM ’12. Association for Computing Machinery, pp. 433–442. DOI: 10.1145/2124295.2124348.

[238] Spiliopoulou, M., M. Bamshad, B. Berendt, and M. Nakagawa (2003). “A Framework for the Evaluation of Session Reconstruction Heuristics in Web Usage Analysis”. In: *Informs Journal on Computing* 15.2, pp. 171–190. DOI: 10.1287/ijoc.15.2.171.14445.

[239] Spiliopoulou, M. and L. C. Faulstich (1999). “WUM: A Tool for Web Utilization Analysis”. In: *The World Wide Web and Databases: Proceedings of the International Workshop on the World Wide Web and Databases.* International Workshop on the World Wide Web and Databases (WebDB98), Valencia, Spain, ed. by P. Atzeni, A. Mendelzon, and G. Mecca. WebDB ’98. Lecture Notes in Computer Science, Vol. 1590. Springer, pp. 184–203. DOI: 10.1007/10704656_12.

[240] Spink, A., B. Jansen, D. Wolfram, and T. Saracevic (2002a). “From E-Sex to e-Commerce: Web Search Changes”. In: *Computer* 35.3, pp. 107–109. DOI: 10.1109/2.989940.

[241] Spink, A. (1996). “Multiple Search Sessions Model of End-User Behavior: An Exploratory Study”. In: *Journal of the American Society for Information Science* 47.8, pp. 603–609. DOI: 10.1002/(SICI)1097-4571(199608)47:8<603::AID-ASI4>3.0.CO;2-X.

[242] Spink, A., B. J. Jansen, and H. Cenk Ozmultu (2000). “Use of Query Reformulation and Relevance Feedback by Excite Users”. In: *Internet Research* 10.4, pp. 317–328. DOI: 10.1108/10662240010342621.

[243] Spink, A., H. C. Ozmutlu, and S. Ozmutlu (2002). “Multitasking Information Seeking and Searching Processes”. In: *Journal of the American Society for Information Science and Technology* 53.8, pp. 639–652. DOI: 10.1002/asi.10124.

[244] Spink, A., S. Ozmutlu, H. C. Ozmutlu, and B. J. Jansen (2002b). “U.S. versus European Web Searching Trends”. In: *ACM SIGIR Forum* 36.2, pp. 32–38. DOI: 10.1145/792550.792555.

[245] Spink, A., M. Park, B. J. Jansen, and J. Pedersen (2006). “Multitasking during Web Search Sessions”. In: *Information Processing & Management* 42.1, pp. 264–275. DOI: 10.1016/j.ipm.2004.10.004.

[246] Spink, A., D. Wolfram, M. B. J. Jansen, and T. Saracevic (2001). “Searching the Web: The Public and Their Queries”. In: *Journal of the American Society for Information Science and Technology* 52.3, pp. 226–234. DOI: 10.1002/1097-4571(2000)9999:9999<::AID-ASI1591>3.0.CO;2-R.

[247] Srivastava, M., A. K. Srivastava, R. Garg, and P. K. Mishra (2017). “Experimental Study of Time Oriented and Referrer Oriented Session Identification Methods in Web Usage Mining”. In: *IJEE* 9.1, pp. 177–183.

[248] Su, N., J. He, Y. Liu, M. Zhang, and S. Ma (2018). “User Intent, Behaviour, and Perceived Satisfaction in Product Search”. In: *Proceedings of the 11th ACM International Conference on Web Search and Data Mining.* 11th ACM International Conference on Web Search and Data Mining (WSDM18), Marina Del Rey, CA, USA, ed. by Y. Chang, C. Zhai, Y. Liu, et al. WSDM ’18. Association for Computing Machinery, pp. 547–555. DOI: 10.1145/3159652.3159714.

[249] Taherdoost, H. (2017). “Determining Sample Size; How to Calculate Survey Sample Size”. In: *International Journal of Economics and Management Systems* 2, p. 3.

[250] Tan, Y. K., X. Xu, and Y. Liu (2016). “Improved Recurrent Neural Networks for Session-based Recommendations”. In: *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems.* 1st Workshop on Deep Learning for Recommender Systems (DLRS16), Boston, MA, USA, ed. by A. Karatzoglou, B. Hidasi, D. Tikk, et al. DLRS ’16. Association for Computing Machinery, pp. 17–22. DOI: 10.1145/2988450.2988452.

[251] Tuan, T. X. and T. M. Phuong (2017). “3D Convolutional Networks for Session-based Recommendation with Content Features”. In: *Proceedings of the Eleventh ACM Conference on Recommender Systems.* Eleventh ACM Conference on Recommender Systems (RecSys17), Como, Italy, ed. by P. Cremonesi, F. Ricci, S. Berkovsky, et al. RecSys ’17. Association for Computing Machinery, pp. 138–146. DOI: 10.1145/3109859.3109900.

[252] Twardowski, B. (2016). “Modelling Contextual Information in Session-Aware Recommender Systems with Neural Networks”. In: *Proceedings of the 10th ACM Conference on Recommender Systems.* 10th ACM Conference on Recommender Systems (RecSys16), Boston, MA, USA, ed. by S. Sen, W. Geyer, J. Freyne, et al. RecSys ’16. Association for Computing Machinery, pp. 273–276. DOI: 10.1145/2959100.2959162.

[253] Uprety, S., Y. Su, D. Song, and J. Li (2018). “Modeling Multidimensional User Relevance in IR Using Vector Spaces”. In: *Proceedings of the 41st International ACM SIGIR Conference on Research & Development in Information Retrieval.* 41st International ACM SIGIR Conference on Research & Development in Information Retrieval (SIGIR18), Ann Arbor, MI, USA, ed. by K. Collins-Thompson, Q. Mei, B. Davison, et al. SIGIR ’18. Association for Computing Machinery, pp. 993–996. DOI: 10.1145/3209978.3210130.

[254] Ustinovskiy, Y., A. Mazur, and P. Serdyukov (2013). “Intent-Based Browse Activity Segmentation”. In: *Advances in Information Retrieval: Proceedings of the 35th European Conference on Information Retrieval Research.* European Conference on Information Retrieval Research (ECIR13), Moscow, Russia, ed. by P. Serdyukov, P. Braslavski, S. O. Kuznetsov, et al. ECIR ’13. Lecture Notes in Computer Science, Vol. 7814. Springer, pp. 242–253. DOI: 10.1007/978-3-642-36973-5_21.

[255] Vassio, L., I. Drago, M. Mellia, Z. B. Houidi, and M. L. Lamali (2018). “You, the Web and Your Device: Longitudinal Characterization of Browsing Habits”. In: *ACM Transactions on the Web* 12.4, pp. 1–30. DOI: 10.1145/3231466.

[256] Vassøy, B. (2018). “Inter-Session Temporal Modeling in Session-Based Recommendation Using Hierarchical Recurrent Neural Networks”. MA thesis. Trondheim, Norway: Norwegian University of Science and Technology. 101 pp.

[257] Vassøy, B., M. Ruocco, E. de Souza da Silva, and E. Aune (2019). “Time Is of the Essence: A Joint Hierarchical RNN and Point Process Model for Time and Item Predictions”. In: *Proceedings of the 12th ACM International Conference on Web Search and Data Mining.* 12th ACM International Conference on Web Search and Data Mining (WSDM19), Melbourne, VIC, Australia, ed. by J. S. Culpepper, A. Moffat, P. N. Bennett, et al. WSDM ’19. Association for Computing Machinery, pp. 591–599. DOI: 10.1145/3289600.3290987.

[258] Verberne, S., M. Sappelli, K. Järvelin, and W. Kraaij (2015). “User Simulations for Interactive Search: Evaluating Personalized Query Suggestion”. In: *Advances in Information Retrieval: Proceedings of the 37th European Conference on Information Retrieval Research.* 37th European Conference on Information Retrieval Research (ECIR2015), Vienna, Austria, ed. by A. Hanbury, G. Kazai, A. Rauber, et al. ECIR ’15. Lecture Notes in Computer Science, Vol. 9022. Springer, pp. 678–690. DOI: 10.1007/978-3-319-16354-3_75.

[259] Verma, M. and E. Yilmaz (2014). “Entity Oriented Task Extraction from Query Logs”. In: *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management.* 23rd ACM International Conference on Conference on Information and Knowledge Management (CIKM14), Shanghai, China, ed. by J. Li, X. S. Wang, M. Garofalakis, et al. CIKM ’14. Association for Computing Machinery, pp. 1975–1978. DOI: 10.1145/2661829.2662076.

[260] — (2016). “Category Oriented Task Extraction”. In: *Proceedings of the 2016 ACM Conference on Human Information Interaction and Retrieval.* 2016 ACM Conference on Human Information Interaction and Retrieval (CHIIR16), Carrboro, NC, USA, ed. by D. Kelly, R. Capra, N. Belkin, et al. CHIIR ’16. Association for Computing Machinery, pp. 333–336. DOI: 10.1145/2854946.2854997.

[261] Völske, M. (2019). “Retrieval Enhancements for Task-Based Web Search”. PhD thesis. Weimar, Germany: Bauhaus-Universität Weimar. 169 pp.

[262] Völske, M., E. Fatehifar, B. Stein, and M. Hagen (2019). “Query-Task Mapping”. In: *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval.* 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR19), Paris, France, ed. by B. Piwowarski, M. Chevalier, Y. Mareek, et al. SIGIR’19. Association for Computing Machinery, pp. 969–972. DOI: 10.1145/3331184.3331286.

[263] Vu, T., A. Willis, S. N. Tran, and D. Song (2015). “Temporal Latent Topic User Profiles for Search Personalisation”. In: *Advances in Information Retrieval: Proceedings of the 37th European Conference on Information Retrieval Research.* 37th European

Conference on Information Retrieval Research (ECIR15), Vienna, Austria, ed. by A. Hanbury, G. Kazai, A. Rauber, et al. ECIR '15. Lecture Notes in Computer Science, Vol. 9022. Springer, pp. 605–616. DOI: 10.1007/978-3-319-16354-3_67.

[264] Wan, M. and J. McAuley (2018). "Item Recommendation on Monotonic Behavior Chains". In: *Proceedings of the 12th ACM Conference on Recommender Systems.* 12th ACM Conference on Recommender Systems (RecSys18), Vancouver, BC, Canada, ed. by S. Pera, M. Ekstrand, X. Amatriain, et al. RecSys '18. Association for Computing Machinery, pp. 86–94. DOI: 10.1145/3240323.3240369.

[265] Wang, H., Y. Song, M.-W. Chang, X. He, A. Hassan, and R. W. White (2014). "Modeling Action-Level Satisfaction for Search Task Satisfaction Prediction". In: *Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval.* 37th International ACM SIGIR Conference on Research & Development in Information Retrieval (SIGIR14), Gold Coast, QLD, Australia, ed. by S. Geva, A. Trotman, P. Bruza, et al. SIGIR '14. Association for Computing Machinery, pp. 123–132. DOI: 10.1145/2600428.2609607.

[266] Wang, H., Y. Song, M.-W. Chang, X. He, R. W. White, and W. Chu (2013). "Learning to Extract Cross-Session Search Tasks". In: *Proceedings of the 22nd International Conference on World Wide Web.* 22nd International Conference on World Wide Web (WWW13), Rio de Janeiro, Brazil, ed. by D. Schwabe, V. Almeida, H. Glaser, et al. WWW '13. Association for Computing Machinery, pp. 1353–1364. DOI: 10.1145/2488388.2488507.

[267] Wang, J., P. Huang, H. Zhao, Z. Zhang, B. Zhao, and D. L. Lee (2018). "Billion-Scale Commodity Embedding for E-commerce Recommendation in Alibaba". In: *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining.* 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD18), London, UK, ed. by Y. Guo and F. Farooq. KDD '18. Association for Computing Machinery, pp. 839–848. DOI: 10.1145/3219819.3219869.

[268] Wang, M., W. Li, and Y. Yan (2019). "Time-Weighted Attentional Session-Aware Recommender System". In: *CoRR.* Arxiv abs/1909.05414.

[269] Wang, T.-X. and W.-H. Lu (2015). "Constructing Complex Search Tasks with Coherent Subtask Search Goals". In: *ACM Transactions on Asian and Low-Resource Language Information Processing* 15.2, pp. 1–29. DOI: 10.1145/2742547.

[270] Wei, C., W. Sen, Z. Yuan, and C. Lian-chang (2009). "Algorithm of Mining Sequential Patterns for Web Personalization Services". In: *ACM SIGMIS Database: the Database for Advances in Information Systems* 40.2, pp. 57–66. DOI: 10.1145/1531817.1531825.

[271] Weissweiler, L. and A. Fraser (2018). "Developing a Stemmer for German Based on a Comparative Analysis of Publicly Available Stemmers". In: *Language Technologies for the Challenges of the Digital Age: Proceedings of the International Conference of the German Society for Computational Linguistics and Language Technology.* International Conference of the German Society for Computational Linguistics and Language Technology (GSCL18), Berlin, Germany, ed. by G. Rehm and T. Declerck. GSCL '18. Lec-

ture Notes in Computer Science, Vol. 10713. Springer, pp. 81–94. DOI: 10.1007/978-3-319-73706-5_8.

[272] White, R. W., P. Bailey, and L. Chen (2009). “Predicting User Interests from Contextual Information”. In: *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval.* 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR09), Boston, MA, USA, ed. by J. Allan, J. Aslam, M. Sanderson, et al. SIGIR ’09. Association for Computing Machinery, pp. 363–370. DOI: 10.1145/1571941.1572005.

[273] White, R. W., P. N. Bennett, and S. T. Dumais (2010). “Predicting Short-Term Interests Using Activity-Based Search Context”. In: *Proceedings of the 19th ACM International Conference on Information and Knowledge Management.* 19th ACM International Conference on Information and Knowledge Management (CIKM10), Toronto, ON, Canada, ed. by J. Huang, N. Koudas, G. Jones, et al. CIKM ’10. Association for Computing Machinery, pp. 1009–1018. DOI: 10.1145/1871437.1871565.

[274] White, R. W., W. Chu, A. Hassan, X. He, Y. Song, and H. Wang (2013). “Enhancing Personalized Search by Mining and Modeling Task Behavior”. In: *Proceedings of the 22nd International Conference on World Wide Web.* 22nd International Conference on World Wide Web (WWW13), Rio de Janeiro, Brazil, ed. by D. Schwabe, V. Almeida, H. Glaser, et al. WWW ’13. Association for Computing Machinery, pp. 1411–1420. DOI: 10.1145/2488388.2488511.

[275] White, R. W. and S. M. Drucker (2007). “Investigating Behavioral Variability in Web Search”. In: *Proceedings of the 16th International Conference on World Wide Web.* 16th International Conference on World Wide Web (WWW07), Banff, AB, Canada, ed. by C. Williamson, M. E. Zurko, P. Patel-Schneider, et al. WWW ’07. Association for Computing Machinery, pp. 21–30. DOI: 10.1145/1242572.1242576.

[276] White, R. W., S. T. Dumais, and J. Teevan (2009). “Characterizing the Influence of Domain Expertise on Web Search Behavior”. In: *Proceedings of the Second ACM International Conference on Web Search and Data Mining.* Second ACM International Conference on Web Search and Data Mining (WSDM09), Barcelona, Spain, ed. by R. Baeza-Yates, P. Boldi, B. Ribeiro-Neto, et al. WSDM ’09. Association for Computing Machinery, pp. 132–141. DOI: 10.1145/1498759.1498819.

[277] White, R. W. and J. Huang (2010). “Assessing the Scenic Route: Measuring the Value of Search Trails in Web Logs”. In: *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval.* 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR10), Geneva, Switzerland, ed. by F. Crestani, S. Marchand-Maillet, H.-H. Chen, et al. SIGIR ’10. Association for Computing Machinery, pp. 587–594. DOI: 10.1145/1835449.1835548.

[278] White, R. W. and D. Morris (2007). “Investigating the Querying and Browsing Behavior of Advanced Search Engine Users”. In: *Proceedings of the 30th International ACM SIGIR Conference on Research and Development in Information Retrieval.* 30th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR07), Amsterdam, Netherlands, ed. by W. Kraaij, A. P. de Vries, C. L. A.

Clarke, et al. SIGIR '07. Association for Computing Machinery, pp. 255–262. DOI: 10.1145/1277741.1277787.

[279] White, R. W., I. Ruthven, and J. M. Jose (2005). "A Study of Factors Affecting the Utility of Implicit Relevance Feedback". In: *Proceedings of the 28th International ACM SIGIR Conference on Research and Development in Information Retrieval.* 28th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR05), Salvador, Brazil, ed. by R. Baeza-Yates, N. Ziviani, G. Marchionini, et al. SIGIR '05. Association for Computing Machinery, pp. 35–42. DOI: 10.1145/1076034.1076044.

[280] Wolfram, D., A. Spink, B. J. Jansen, and T. Saracevic (2001). "Vox Populi: The Public Searching of the Web". In: *Journal of the American Society for Information Science and Technology* 52.12, pp. 1073–1074. DOI: 10.1002/asi.1157.

[281] Wolfram, D., P. Wang, and J. Zhang (2007). "Modeling Web Session Behavior Using Cluster Analysis: A Comparison of Three Search Settings". In: *Proceedings of the American Society for Information Science and Technology* 44.1, pp. 1–13. DOI: 10.1002/meet.1450440232.

[282] — (2009). "Identifying Web Search Session Patterns Using Cluster Analysis: A Comparison of Three Search Environments". In: *Journal of the American Society for Information Science and Technology* 60.5, pp. 896–910. DOI: 10.1002/asi.21034.

[283] Wu, C., M. Yan, and L. Si (2017). "Session-Aware Information Embedding for E-commerce Product Recommendation". In: *Proceedings of the 2017 ACM Conference on Information and Knowledge Management.* 2017 ACM Conference on Information and Knowledge Management (CIKM17), Singapore, Singapore, ed. by E.-P. Lim, M. Winslett, M. Sanderson, et al. CIKM '17. Association for Computing Machinery, pp. 2379–2382. DOI: 10.1145/3132847.3133163.

[284] Wu, D., J. Dong, and Y. Tang (2018). "Modeling and Analyzing Information Preparation Behaviors in Cross-Device Search". In: *Cross-Cultural Design. Methods, Tools, and Users: Proceedings of the 10th International Conference on Cross-Cultural Design.* 10th International Conference on Cross-Cultural Design (CCD18), Las Vegas, NV, USA, ed. by P.-L. P. Rau. CCD '18. Lecture Notes in Computer Science, Vol. 10911. Springer, pp. 232–249. DOI: 10.1007/978-3-319-92141-9_18.

[285] Wu, L., D. Hu, L. Hong, and H. Liu (2018). "Turning Clicks into Purchases: Revenue Optimization for Product Search in E-Commerce". In: *Proceedings of the 41st International ACM SIGIR Conference on Research & Development in Information Retrieval.* 41st International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR18), Ann Arbor, MI, USA, ed. by K. Collins-Thompson, Q. Mei, B. Davison, et al. SIGIR '18. Association for Computing Machinery, pp. 365–374. DOI: 10.1145/3209978.3209993.

[286] Xiang, B., D. Jiang, J. Pei, X. Sun, E. Chen, and H. Li (2010). "Context-Aware Ranking in Web Search". In: *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval.* 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR10), Geneva,

Switzerland, ed. by F. Crestani, S. Marchand-Maillet, H.-H. Chen, et al. SIGIR '10. Association for Computing Machinery, pp. 451–458. DOI: 10.1145/1835449.1835525.

[287] Yang, Z. and E. Nyberg (2015). "Leveraging Procedural Knowledge for Task-oriented Search". In: *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval.* 38th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR15), Santiago, Chile, ed. by R. Baeza-Yates, M. Lalmas, A. Moffat, et al. SIGIR '15. Association for Computing Machinery, pp. 513–522. DOI: 10.1145/2766462.2767744.

[288] Yao, Q., X. Huang, and A. An (2006). "Applying Language Modeling to Session Identification from Database Trace Logs". In: *Knowledge and Information Systems* 10.4, pp. 473–504. DOI: 10.1007/s10115-006-0015-9.

[289] Ye, C. and M. L. Wilson (2014). "A User Defined Taxonomy of Factors That Divide Online Information Retrieval Sessions". In: *Proceedings of the 5th Information Interaction in Context Symposium.* 5th Information Interaction in Context Symposium (IIiX14), Regensburg, Germany, ed. by D. Elsweiler, B. Ludwig, L. Azzopardi, et al. IIiX '14. Association for Computing Machinery, pp. 48–57. DOI: 10.1145/2637002.2637010.

[290] Yu, R., U. Gadiraju, P. Holtz, M. Rokicki, P. Kemkes, and S. Dietze (2018). "Predicting User Knowledge Gain in Informational Search Sessions". In: *Proceedings of the 41st International ACM SIGIR Conference on Research & Development in Information Retrieval.* 41st International ACM SIGIR Conference on Research & Development in Information Retrieval (SIGIR18), Ann Arbor, MI, USA, ed. by K. Collins-Thompson, Q. Mei, B. Davison, et al. SIGIR '18. Association for Computing Machinery, pp. 75–84. DOI: 10.1145/3209978.3210064.

[291] Yuankang, F. and H. Zhiqiu (2010). "A Session Identification Algorithm Based on Frame Page and Pagethreshold". In: *2010 3rd International Conference on Computer Science and Information Technology.* 3rd International Conference on Computer Science and Information Technology (ICCSIT10), Chengdu, China. ICCSIT '10. IEEE, pp. 645–647. DOI: 10.1109/ICCSIT.2010.5564697.

[292] Žaloudek, J. (2018). "User Behaviour Clustering and Behaviour Modeling Based on Clickstream Data". MA thesis. Prague, Czech Republic: Czech Technical University in Prague.

[293] Zhang, H., X. Song, C. Xiong, C. Rosset, P. N. Bennett, N. Craswell, and S. Tiwary (2019). "Generic Intent Representation in Web Search". In: *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval.* 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR19), Paris, France, ed. by B. Piwowarski, M. Chevalier, E. Gaussier, et al. SIGIR '19. Association for Computing Machinery, pp. 65–74. DOI: 10.1145/3331184.3331198.

[294] Zhao, Q., M. C. Willemsen, G. Adomavicius, F. M. Harper, and J. A. Konstan (2019). "From Preference into Decision Making: Modeling User Interactions in Recommender Systems". In: *Proceedings of the 13th ACM Conference on Recommender Systems.* 13th ACM Conference on Recommender Systems (RecSys19), Copenhagen, Denmark, ed.

by T. Bogers, A. Said, P. Brusilovsky, et al. RecSys '19. Association for Computing Machinery, pp. 29–33. DOI: 10.1145/3298689.3347065.

[295] Zhou, B., S. C. Hui, and A. C. Fong (2006). "An Effective Approach for Periodic Web Personalization". In: *Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence (Main Conference).* 2006 IEEE/WIC/ACM International Conference on Web Intelligence (WI06), Hong Kong, Hong Kong, ed. by T. Nishida, Z. Shi, U. Visser, et al. WI '06. IEEE, pp. 284–292. DOI: 10.1109/WI.2006.36.

[296] Zhou, Q., H. Ye, and Z. Ding (2012). "Performance Analysis of Web Applications Based on User Navigation". In: *Physics Procedia* 24, pp. 1319–1328. DOI: 10.1016/j.phpro.2012.02.197.

[297] Zhou, X., P. Zhang, and J. Wang (2016a). "Examining Task Relationships in Multitasking Consumer Search Sessions: A Query Log Analysis". In: *Proceedings of the Association for Information Science and Technology* 53.1, pp. 1–5. DOI: 10.1002/pra2.2016.14505301102.

[298] — (2016b). "Identification and Analysis of Multi-tasking Product Information Search Sessions with Query Logs". In: *Journal of Data and Information Science* 1.3, pp. 79–94. DOI: 10.20309/jdis.201621.

[299] Zhu, Y., E. Yan, and F. Wang (2017). "Semantic Relatedness and Similarity of Biomedical Terms: Examining the Effects of Recency, Size, and Section of Biomedical Publications on the Performance of Word2vec". In: *BMC Medical Informatics and Decision Making* 17.95. DOI: 10.1186/s12911-017-0498-1.

[300] Zhu, Y., H. Li, Y. Liao, B. Wang, Z. Guan, H. Liu, and D. Cai (2017). "What to Do Next: Modeling User Behaviors by Time-LSTM". In: *Proceedings of the 26th International Joint Conference on Artificial Intelligence.* 26th International Joint Conference on Artificial Intelligence (IJCAI17), Melbourne, VIC, Australia, ed. by C. Sierra. IJCAI '17. AAAI Press, pp. 3602–3608. DOI: 10.5555/3172077.3172393.

[301] Zhuang, M., G. Demartini, and E. G. Toms (2017). "Understanding Engagement through Search Behaviour". In: *Proceedings of the 2017 ACM Conference on Information and Knowledge Management.* 2017 ACM Conference on Information and Knowledge Management (CIKM17), Singapore, Singapore, ed. by E.-P. Lim, M. Winslett, M. Sanderson, et al. CIKM '17. Association for Computing Machinery, pp. 1957–1966. DOI: 10.1145/3132847.3132978.

Sources

[1] ACM RecSys Challenge 2016. http://2016.recsyschallenge.com/. Retrieved 15 November 2021.

[2] AOL Search Engine. https://www.aol.de/. Retrieved 15 November 2021.

[3] AOL data leak on Wikipedia. https://en.wikipedia.org/wiki/AOL_search_data_leak. Retrieved 15 November 2021.

[4] API Reference for HDBSCAN. https://hdbscan.readthedocs.io/en/latest/api.html. Retrieved 28 November 2021.

[5] Accepted Workshops ACM WSDM Conference 2014. http://www.wsdm-conference.org/2014/accepted-workshops/. Retrieved 15 November 2021.

[6] Amazon Athena. https://aws.amazon.com/de/athena/. Retrieved 15 November 2021.

[7] Amazon EMR. https://aws.amazon.com/de/emr/. Retrieved 15 November 2021.

[8] Amazon S3. https://aws.amazon.com/de/s3/. Retrieved 15 November 2021.

[9] Amazon Web Services. https://aws.amazon.com/. Retrieved 15 November 2021.

[10] Apache Lucene. https://lucene.apache.org/. Retrieved 15 November 2021.

[11] Apple Siri. https://www.apple.com/uk/siri/. Retrieved 28 November 2021.

[12] Archived version of DMOZ. https://dmoz-odp.org/. Retrieved 8 December 2021.

[13] Campaign Data in URLs in Google Analytics. https://support.google.com/analytics/answer/1033863. Retrieved 15 November 2021.

[14] Cistem Stemmer Documentation. https://www.nltk.org/_modules/nltk/stem/cistem.html. Retrieved 10 December 2020.

[15] Count of Sessions in Google Analytics. https://support.google.com/analytics/answer/1032796. Retrieved 28 November 2021.

[16] Dogpile Search Engine. https://www.dogpile.com/. Retrieved 5 January 2021.

[17] Ebay Homepage. https://www.ebay.com/. Retrieved 5 January 2022.

[18] Excite Search Engine. http://www.excite.com/. Retrieved 5 January 2021.

[19] FastText library for fast text representation and classification. https://github.com/facebookresearch/fastText. Retrieved 9 December 2021.

[20] Filtered Product Category Page for 4k-Fernseher (4k tvs). https://www.idealo.de/preisvergleich/ProductCategory/4012F1921183.html. Retrieved 15 November 2021.

[21] Filtered Product Category Page for Gibson Les Paul Standard E-Gitarren (e-guitars). https://www.idealo.de/preisvergleich/ProductCategory/5666F1499900.html. Retrieved 15 November 2021.

[22] Filtered Product Category Page for Withings Personenwaagen (personal scales). https://www.idealo.de/preisvergleich/ProductCategory/3933F1451777.html. Retrieved 15 November 2021.

[23] Fingerprinting to Track Us Online. https://www.nytimes.com/2019/07/03/technology/personaltech/fingerprinting-track-devices-what-to-do.html. Retrieved 15 November 2021.

[24] Gensim Python Library. https://radimrehurek.com/gensim/. Retrieved 28 November 2021.

[25] Gensim word2vec Implementation. https://radimrehurek.com/gensim_3.8.3/models/word2vec.html. Retrieved 28 November 2021.

[26] German Stopword List. https://solariz.de. Retrieved 10 December 2020. No longer accessible.

[27] HTTP cookie on Wikipedia. https://en.wikipedia.org/wiki/HTTP_cookie. Retrieved 15 November 2021.

[28] IP address on Wikipedia. https://en.wikipedia.org/wiki/IP_address. Retrieved 1 December 2021.

[29] Implementation of HDBSCAN clustering. https://github.com/scikit-learn-contrib/hdbscan. Retrieved 28 November 2021.

[30] Inter-/Intra-session Recurrent Neural Network for Session-based Recommender Systems. https://github.com/olesls/master_thesis. Retrieved 28 November 2021.

[31] Last.fm Homepage. https://www.last.fm/. Retrieved 1 December 2021.

[32] Microsoft Cortana. https://support.microsoft.com/en-us/topic/what-is-cortana-953e648d-5668-e017-1341-7f26f7d0f825. Retrieved 28 November 2021.

[33] MinMaxScaler in scikit-learn. https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html. Retrieved 28 November 2021.

[34] Personalized Web Search Challenge at Kaggle. https://www.kaggle.com/c/yandex-personalized-web-search-challenge/overview. Retrieved 15 November 2021.

[35] Presto SQL Engine. https://prestodb.io/. Retrieved 15 November 2021.

[36] Probase at Microsoft. https://www.microsoft.com/en-us/research/project/probase/. Retrieved 9 November 2021.

[37] Project Jupyter. https://jupyter.org/. Retrieved 15 November 2021.

[38] Reddit Homepage. https://www.reddit.com/. Retrieved 5 December 2021.

[39] scikit-learn Python Library. https://scikit-learn.org/stable/. Retrieved 28 November 2021.

[40] Select documentation for AWS Athena. https://docs.aws.amazon.com/athena/latest/ug/select.html. Retrieved 28 November 2021.

[41] Session Definition in Google Analytics. https://support.google.com/analytics/answer/2731565. Retrieved 8 June 2020.

[42] Silhouette-Coefficient in scikit-learn. https://scikit-learn.org/stable/modules/clustering.html#silhouette-coefficient. Retrieved 1 December 2021.

[43] Sowiport Database for Social Sciences. https://www.hbk-bs.de/einrichtungen/bibliothek/recherche/fachdatenbanken-alphabetisch/sowiport-csa-sozialwissenschaftliche-datenbanken/index.php. Retrieved 1 December 2021.

[44] Splunk Homepage. https://www.splunk.com. Retrieved 15 November 2021.

[45] Standard Score on Wikipedia. https://en.wikipedia.org/wiki/Standard_score. Retrieved 28 November 2021.

[46] Taobao Homepage. https://world.taobao.com/. Retrieved 5 January 2022.

[47] TensorFlow Homepage. https://www.tensorflow.org/. Retrieved 28 November 2021.

[48] Text Retrieval Conference. https://trec.nist.gov/. Retrieved 15 November 2021.

[49] The TianGong-ST Dataset. http://www.thuir.cn/tiangong-st/. Retrieved 15 November 2021.

[50] WikiHow Homepage. https://www.wikihow.com/. Retrieved 15 November 2021.

[51] XING Homepage. https://www.xing.com/. Retrieved 15 November 2021.

[52] Yahoo Search Engine. https://de.yahoo.com/. Retrieved 15 November 2021.

[53] Yandex Search Engine. https://yandex.com/. Retrieved 15 November 2021.

[54] YouTube Homepage. https://www.youtube.com/. Retrieved 15 November 2021.

[55] idealo About Us Page. https://www.idealo.de/unternehmen/ueber-uns/. Retrieved 1 November 2021.

[56] idealo Price Comparison. https://www.idealo.de/. Retrieved 10 December 2021.

Appendices

A1 Overview of the columns in the dataset

Field	Description	Datatype
browser	Name of the browser used for the interaction	string(255)
category_id	Identifier of the category of the page interacted with	bigint(19)
category_name	Name of the category of the page interacted with	string(255)
category_synonyms	Synonyms of the category of the page interacted with	string(255)
category_type	Type of the category of the page interacted with	string(255)
cluster_id	Id of the cluster page interacted with	string(255)
compared_products	List of product_ids used in the product comparison user action	string(255)
device	Type of device used for the interaction	string(255)
http_referer	URL of the previous page visited by a user	string(255)
interaction$_{day}$	Human-readable date of the interaction	date(10)
last_tracetime	Timestamp of the (chronologically) previous interaction of a user	bigint(19)
list_id	Id of the list page interacted with	string(255)
list_query	Query used to generate the list interacted with	string(255)
main_product_id	Id of the parent product of a product_id when existing, otherwise product_id	bigint(19)
man_query	Query used to search for manufacturers on the respective manufacturer pages	string(255)
manufacturer_id	Identifier of the manufacturer of a visited product	string(255)
number_of_interactions	Total of all interactions by an individual user entity	bigint(19)
offer_id	Identifier of the offer interacted with	string(255)
orders	Identifier to mark an order	bigint(19)
os	Type of operating system used to visit the page	string(255)
page_template	Type description of the page visited by a user	string(255)
parent_category_id	Identifier of the parent category of the category interacted with	string(255)
product_id	Identifier of a visited product	bigint(19)
product_name	Name of a visited product	string(255)
product_types	Type of a visited product	string(255)
query	Query issued by a user	string(255)
referer_list_query	List query extracted from the referer	string(255)
referer_query	Query issued by a user extracted from the referer	string(255)
root_category_id	Identifier of the highest level of a visited category	bigint(19)
root_category_name	Name of the highest level of a visited category	string(255)
shop_id	Identifier for the shop a user interacted with	string(255)
shop_query	Query used to search for shops on the respective shop pages	string(255)
timespan	Time between subsequent interactions of a user	double(53)
trace_id	Unique identifier for every interaction	string(255)
tracetime	Unix timestamp of an interaction in milliseconds	bigint(19)
url	URL of the page visited by a user	string(255)
user_id	General numeric identifier for a user entity	bigint(19)

Table A1: Overview of the columns in the dataset.

A2 Overview of all tested approaches

	Approach	Method	Mechanic	Variant	Comparison context	Identifier
1	M	Structural	Path-based	connecting http_referer and url	Cons., complH.	visit_id
2	M	Temporal	(Fixed) Inactivity Timeout	5m	Cons., dir.	ti5
3	M	Temporal	(Fixed) Inactivity Timeout	10m	Cons., dir.	ti10
4	M	Temporal	(Fixed) Inactivity Timeout	15m	Cons., dir.	ti15
5	M	Temporal	(Fixed) Inactivity Timeout	25.5m	Cons., dir.	ti25
6	M	Temporal	(Fixed) Inactivity Timeout	30m	Cons., dir.	ti30
7	M	Temporal	(Fixed) Inactivity Timeout	45m	Cons., dir.	ti45
8	M	Temporal	(Fixed) Inactivity Timeout	60m	Cons., dir.	ti60
9	M	Temporal	(Fixed) Inactivity Timeout	90m	Cons., dir.	ti90
10	M	Temporal	(Fixed) Inactivity Timeout	120m	Cons., dir.	ti120
11	M	Temporal	(Fixed) Inactivity Timeout	180m	Cons., dir.	ti180
12	M	Temporal	(Fixed) Inactivity Timeout	360m	Cons., dir.	ti360
13	M	Temporal	(Fixed) Inactivity Timeout	720m	Cons., dir.	ti720
14	M	Temporal	(Fixed) Inactivity Timeout	1,440m	Cons., dir.	ti1440
15	M	Temporal	Fixed Length	5m	Cons., dir.	tf5
16	M	Temporal	Fixed Length	10m	Cons., dir.	tf10
17	M	Temporal	Fixed Length	15m	Cons., dir.	tf15
18	M	Temporal	Fixed Length	20m	Cons., dir.	tf20
19	M	Temporal	Fixed Length	30m	Cons., dir.	tf30
20	M	Temporal	Fixed Length	45m	Cons., dir.	tf45
21	M	Temporal	Fixed Length	60m	Cons., dir.	tf60
22	M	Temporal	Fixed Length	90m	Cons., dir.	tf90
23	M	Temporal	Fixed Length	120m	Cons., dir.	tf120

	Approach	Method	Mechanic	Variant	Comparison context	Identifier
24	M	Temporal	Fixed Length	180m	Cons., dir.	tf180
25	M	Temporal	Fixed Length	360m	Cons., dir.	tf360
26	M	Temporal	Fixed Length	720m	Cons., dir.	tf720
27	M	Temporal	Fixed Length	1,440m	Cons., dir.	tf1440
28	M	Temporal	Fixed Length	Session Day	Cons., dir.	tfd
29	M	Temporal	(Dynamic) Inactivity Timeout	per page_template	Cons., dir.	tdp
30	M	Temporal	(Dynamic) Inactivity Timeout	per category_id	Cons., dir.	tdc
31	M	Temporal	(Dynamic) Inactivity Timeout	per root_category_id	Cons., dir.	tdr
32	M	Temporal	(Dynamic) Inactivity Timeout	per page_template, category_id	Cons., dir.	tdpc
33	M	Temporal	(Dynamic) Inactivity Timeout	per page_template, root_category_id	Cons., dir.	tdpr
34	M	Temporal	(Dynamic) Inactivity Timeout	per page_template, month of interaction_day	Cons., dir.	tdpm
35	M	Temporal	(Dynamic) Inactivity Timeout	per category_id, month of interaction_day	Cons., dir.	tdcm
36	M	Temporal	(Dynamic) Inactivity Timeout	per page_template, device	Cons., dir.	tdpd
37	M	Temporal	(Dynamic) Inactivity Timeout	per page_template, category_id and device	Cons., dir.	tdpcd
38	L	Lexical	Matching	category_ids, root_category_ids	Cons., dir.	lcdb1
39	L	Lexical	Matching	category_ids, root_category_ids	All, dir.	ladb1
40	L	Semantic	Term Space	sim. category_ids according to bm25L ranking (top 10)	Cons., complH.	bm25cc
41	L	Semantic	Term Space	sim. category_ids according to bm25L ranking (top 10)	Cons., dir.	bm25cd
42	L	Semantic	Term Space	sim. category_ids according to bm25L ranking (top 10)	All, complH.	bm25ac
43	L	Semantic	Term Space	sim. category_ids according to bm25L ranking (top 10)	All, dir.	bm25ad
44	L	Semantic	userCat2Vec	cosine sim. category_ids (top 10)	Cons., complH.	u2v10cc
45	L	Semantic	userCat2Vec	cosine sim. category_ids (top 10)	Cons., dir.	u2v10cd
46	L	Semantic	userCat2Vec	cosine sim. category_ids (top 10)	All, complH.	u2v10ac
47	L	Semantic	userCat2Vec	cosine sim. category_ids (top 10)	All, dir.	u2v10ad
48	L	Semantic	userCat2Vec	cosine sim. category_ids (> 0.5)	Cons., complH.	u2v05cc
49	L	Semantic	userCat2Vec	cosine sim. category_ids (> 0.5)	Cons., dir.	u2v05cd
50	L	Semantic	userCat2Vec	cosine sim. category_ids (> 0.5)	All, complH.	u2v05ac
51	L	Semantic	userCat2Vec	cosine sim. category_ids (> 0.5)	All, dir.	u2v05ad
52	L	Semantic	userCat2Vec	cosine sim. category_ids (cutoff)	Cons., complH.	u2vccc
53	L	Semantic	userCat2Vec	cosine sim. category_ids (cutoff)	Cons., dir.	u2vccd
54	L	Semantic	userCat2Vec	cosine sim. category_ids (cutoff)	All, complH.	u2vcac
55	L	Semantic	userCat2Vec	cosine sim. category_ids (cutoff)	All, dir.	u2vcad
56	C	Geometric	userCat2Vec	cosine sim., 24h	Cons., complH.	geomu24cc
57	C	Geometric	userCat2Vec	cosine sim., 24h	Cons., dir.	geomu24cd
58	C	Geometric	userCat2Vec	cosine sim., 14d	Cons., complH.	geomu14cc
59	C	Geometric	userCat2Vec	cosine sim., 14d	Cons., dir.	geomu14cd
60	C	Geometric	userCat2Vec	cosine sim., 24h	All, complH.	geomu24ac
61	C	Geometric	userCat2Vec	cosine sim., 24h	All, dir.	geomu24ad
62	C	Geometric	userCat2Vec	cosine sim., 14d	All, complH.	geomu14ac
63	C	Geometric	userCat2Vec	cosine sim., 14d	All, dir.	geomu14ad
64	C	Geometric	userCat2Vec	cosine sim., 75d	All, complH.	geomu75ac
65	C	Geometric	userCat2Vec	cosine sim., 75d	All, dir.	geomu75ad
66	C	Lexical, temporal	matching, inactivity	category_id, root_category_id, 5m	Cons., dir.	lti5cdb1

	Approach	Method	Mechanic	Variant	Comparison context	Identifier
67	C	Lexical, temporal	matching, inactivity	category_id, root_category_id, 30m	Cons., dir.	lti30cdb1
68	C	Lexical, temporal	matching, inactivity	category_id, root_category_id, 1,440m	Cons., dir.	lti1cdb1
69	C	Lexical, temporal	matching, inactivity	category_id, root_category_id, 14d	Cons, dir.	lti14cdb1
70	C	Lexical, temporal	matching, inactivity	category_id, root_category_id, 1d	All, dir.	lti1adb1
71	C	Lexical, temporal	matching, inactivity	category_id, root_category_id, 14d	All, dir.	lti14adb1
72	C	Lexical, temporal	matching, inactivity	category_id, root_category_id, 75d	All, dir.	lti75adb1
73	C	Lexical, temporal	matching, inactivity	category_id, root_category_id, 180d	All, dir.	lti180adb1
74	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. top 10, 5m	Cons., complH.	u2v10ti5cc
75	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. top 10, 5m	Cons., dir.	u2v10ti5cd
76	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. > 0.5, 5m	Cons., complH.	u2v05ti5cc
77	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. > 0.5, 5m	Cons., dir.	u2v05ti5cd
78	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. cutoff, 5m	Cons., complH.	u2vcti5cc
79	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. cutoff, 5m	Cons., dir.	u2vcti5cd
80	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. top 10, 30m	Cons., complH.	u2v10ti30cc
81	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. top 10, 30m	Cons., dir.	u2v10ti30cd
82	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. > 0.5, 30m	Cons., complH.	u2v05ti30cc
83	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. > 0.5, 30m	Cons., dir.	u2v05ti30cd
84	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. cutoff, 30m	Cons., complH.	u2vcti30cc
85	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. cutoff, 30m	Cons., dir.	u2vcti30cd
86	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. top 10, 1,440m	Cons., complH.	u2v10ti1cc
87	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. top 10, 1,440m	Cons., dir.	u2v10ti1cd
88	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. > 0.5, 1,440m	Cons., complH.	u2v05ti1cc
89	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. > 0.5, 1,440m	Cons., dir.	u2v05ti1cd
90	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. cutoff, 1,440m	Cons., complH.	u2vcti1cc
91	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. cutoff, 1,440m	Cons., dir.	u2vcti1cd
92	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. top 10, 14d	Cons., complH.	u2v10ti14cc
93	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. top 10, 14d	Cons., dir.	u2v10ti14cd
94	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. > 0.5, 14d	Cons., complH.	u2v05ti14cc
95	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. > 0.5, 14d	Cons., dir.	u2v05ti14cd
96	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. cutoff, 14d	Cons., complH.	u2vcti14cc
97	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. cutoff, 14d	Cons., dir.	u2vcti14cd
98	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. top 10, 1d	All, complH.	u2v10ti1ac
99	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. top 10, 1d	All, dir.	u2v10ti1ad
100	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. > 0.5, 1d	All, complH.	u2v05ti1ac
101	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. > 0.5, 1d	All, dir.	u2v05ti1ad
102	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. cutoff, 1d	All, complH.	u2vcti1ac
103	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. cutoff, 1d	All, dir.	u2vcti1ad
104	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. top 10, 14d	All, complH.	u2v10ti14ac
105	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. top 10, 14d	All, dir.	u2v10ti14ad
106	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. > 0.5, 14d	All, complH.	u2v05ti14ac
107	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. > 0.5, 14d	All, dir.	u2v05ti14ad
108	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. cutoff, 14d	All, complH.	u2vcti14ac
109	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. cutoff, 14d	All, dir.	u2vcti14ad
110	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. top 10, 75d	All, complH.	u2v10ti75ac
111	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. top 10, 75d	All, dir.	u2v10ti75ad
112	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. > 0.5, 75d	All, complH.	u2v05ti75ac
113	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. > 0.5, 75d	All, dir.	u2v05ti75ad
114	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. cutoff, 75d	All, complH.	u2vcti75ac
115	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. cutoff, 75d	All, dir.	u2vcti75ad
116	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. top 10, 180d	All, complH.	u2v10ti180ac
117	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. top 10, 180d	All, dir.	u2v10ti180ad
118	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. > 0.5, 180d	All, complH.	u2v05ti180ac
119	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. > 0.5, 180d	All, dir.	u2v05ti180ad
120	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. cutoff, 180d	All, complH.	u2vcti180ac
121	C	Semantic, temporal	usercat2vec, inactivity	cosine sim. cutoff, 180d	All, dir.	u2vcti180ad
122	C	Semantic, temporal	Term space, inactivity	bm25L ranking sim. top 10, 5m	Cons., complH.	bm25ti5cc
123	C	Semantic, temporal	Term space, inactivity	bm25L ranking sim. top 10, 5m	Cons., dir.	bm25ti5cd
124	C	Semantic, temporal	Term space, inactivity	bm25L ranking sim. top 10, 30m	Cons., complH.	bm25ti30cc
125	C	Semantic, temporal	Term space, inactivity	bm25L ranking sim. top 10, 30m	Cons., dir.	bm25ti30cd
126	C	Semantic, temporal	Term space, inactivity	bm25L ranking sim. top 10, 1,440m	Cons., complH.	bm25ti1cc
127	C	Semantic, temporal	Term space, inactivity	bm25L ranking sim. top 10, 1,440m	Cons., dir.	bm25ti1cd

	Approach	Method	Mechanic	Variant	Comparison context	Identifier
128	C	Semantic, temporal	Term space, inactivity	bm25L ranking sim. top 10, 1d	All, complH.	bm25ti1ac
129	C	Semantic, temporal	Term space, inactivity	bm25L ranking sim. top 10, 1d	All, dir.	bm25ti1ad
130	C	Semantic, temporal	Term space, inactivity	bm25L ranking sim. top 10, 14d	All, complH.	bm25ti14ac
131	C	Semantic, temporal	Term space, inactivity	bm25L ranking sim. top 10, 14d	All, dir.	bm25ti14ad
132	C	Semantic, temporal	Term space, inactivity	bm25L ranking sim. top 10, 75d	All, complH.	bm25ti75ac
133	C	Semantic, temporal	Term space, inactivity	bm25L ranking sim. top 10, 75d	All, dir.	bm25ti75ad
134	C	Semantic, temporal	Term space, inactivity	bm25L ranking sim. top 10, 180d	All, complH.	bm25ti180ac
135	C	Semantic, temporal	Term space, inactivity	bm25L ranking sim. top 10, 180d	All, dir.	bm25ti180ad

Table A2: Overview of all tested approaches. Abbreviations: S = Structural, M = Mechanical, L = Logical, C = Combination, Cons. = Consecutive, complH. = complete history, dir. = direct, sim. = Similarity, m = minutes, h = hours, d = days.

A3 Top 5 sequences for session approaches

Interaction Bucket	Sequences	Totals
tf5	li_oop (17.74%) — li_q (5.79%) — li_oop→lo (4.56%) — oop (3.41%) — li_pc (3.13%)	34.63%
tf10	li_oop (17.96%) — li_q (5.92%) — li_oop→lo (4.42%) — oop (3.13%) — li_pc (3.1%)	34.53%
tf15	li_oop (17.95%) — li_q (5.95%) — li_oop→lo (4.33%) — li_pc (3.09%) — oop (2.99%)	34.31%
tf20	li_oop (17.89%) — li_q (5.95%) — li_oop→lo (4.27%) — li_pc (3.08%) — oop (2.9%)	34.09%
tf30	li_oop (17.77%) — li_q (5.94%) — li_oop→lo (4.19%) — li_pc (3.07%) — oop (2.76%)	33.73%
tf45	li_oop (17.6%) — li_q (5.93%) — li_oop→lo (4.12%) — li_pc (3.06%) — oop (2.62%)	33.33%
tf60	li_oop (17.46%) — li_q (5.92%) — li_oop→lo (4.08%) — li_pc (3.05%) — li_oop→li_oop (2.56%)	33.07%
tf90	li_oop (17.22%) — li_q (5.9%) — li_oop→lo (4.02%) — li_pc (3.04%) — li_oop→li_oop (2.64%)	32.82%
tf120	li_oop (17.02%) — li_q (5.88%) — li_oop→lo (3.98%) — li_pc (3.03%) — li_oop→li_oop (2.69%)	32.6%
tf180	li_oop (16.71%) — li_q (5.86%) — li_oop→lo (3.92%) — li_pc (3.01%) — li_oop→li_oop (2.76%)	32.26%
tf360	li_oop (16.15%) — li_q (5.82%) — li_oop→lo (3.81%) — li_pc (2.99%) — li_oop→li_oop (2.87%)	31.64%
tf720	li_oop (15.47%) — li_q (5.76%) — li_oop→lo (3.69%) — li_oop→li_oop (2.98%) — li_pc (2.95%)	30.85%
tf1440	li_oop (14.28%) — li_q (5.58%) — li_oop→lo (3.43%) — li_oop→li_oop (3.11%) — li_pc (2.87%)	29.27%
tfd	li_oop (15.56%) — li_q (5.8%) — li_oop→lo (3.71%) — li_pc (2.98%) — li_oop→li_oop (2.92%)	30.97%
ti5	li_oop (19.12%) — li_q (6.22%) — li_oop, lo (4.76%) — oop (3.35%) — li_pc (3.3%)	36.74%
ti10	li_oop (18.77%) — li_q (6.18%) — li_oop, lo (4.54%) — li_pc (3.22%) — oop (3.09%)	35.8%
ti15	li_oop (18.52%) — li_q (6.13%) — li_oop, lo (4.41%) — li_pc (3.18%) — oop (2.96%)	35.2%
ti26	li_oop (18.16%) — li_q (6.07%) — li_oop, lo (4.27%) — li_pc (3.14%) — oop (2.79%)	34.43%
ti30	li_oop (18.07%) — li_q (6.05%) — li_oop, lo (4.23%) — li_pc (3.13%) — oop (2.74%)	34.22%
ti45	li_oop (17.81%) — li_q (6.01%) — li_oop, lo (4.16%) — li_pc (3.11%) — oop (2.59%)	33.68%
ti60	li_oop (17.63%) — li_q (5.98%) — li_oop, lo (4.11%) — li_pc (3.09%) — li_oop, li_oop (2.53%)	33.34%
ti90	li_oop (17.35%) — li_q (5.95%) — li_oop, lo (4.05%) — li_pc (3.07%) — li_oop, li_oop (2.6%)	33.02%
ti120	li_oop (17.14%) — li_q (5.93%) — li_oop, lo (4.0%) — li_pc (3.06%) — li_oop, li_oop (2.65%)	32.78%
ti180	li_oop (16.82%) — li_q (5.91%) — li_oop, lo (3.94%) — li_pc (3.04%) — li_oop, li_oop (2.72%)	32.43%
ti360	li_oop (16.22%) — li_q (5.87%) — li_oop, lo (3.83%) — li_pc (3.01%) — li_oop, li_oop (2.82%)	31.75%
ti720	li_oop (15.56%) — li_q (5.82%) — li_oop, lo (3.72%) — li_pc (2.99%) — li_oop, li_oop (2.91%)	31.0%
ti1440	li_oop (14.6%) — li_q (5.76%) — li_oop, lo (3.51%) — li_oop, li_oop (2.98%) — li_pc (2.97%)	29.82%
lcdb1	li_oop (9.13%) — li_q (6.83%) — q (3.2%) — li_oop, li_oop (2.51%) — li_pc (2.21%)	23.88%
ladb1	li_oop (7.1%) — li_q (5.42%) — li_oop, li_oop (2.8%) — li_pc (1.88%) — li_hp (1.7%)	18.9%
bm25cd	li_oop (10.71%) — li_q (7.67%) — q (3.88%) — lo (2.69%) — li_pc (2.61%)	27.56%
bm25cc	li_oop (10.9%) — li_q (7.82%) — q (3.87%) — lo (2.68%) — li_pc (2.64%)	27.91%
bm25ad	li_oop (9.24%) — li_q (7.09%) — li_pc (2.6%) — li_oop, li_oop (2.48%) — li_oop, lo (2.25%)	23.66%
bm25ac	li_oop (9.67%) — li_q (7.43%) — li_pc (2.72%) — li_oop, li_oop (2.54%) — li_oop, lo (2.36%)	24.72%
u2v05cd	li_oop (10.93%) — li_q (7.82%) — q (4.22%) — lo (2.79%) — oop (2.64%)	28.4%
u2v05cc	li_oop (10.99%) — li_q (7.86%) — q (4.19%) — lo (2.77%) — li_pc (2.64%)	28.45%
u2v05ad	li_oop (9.7%) — li_q (7.37%) — li_pc (2.71%) — li_oop, li_oop (2.54%) — q (2.48%)	24.8%
u2v05ac	li_oop (9.82%) — li_q (7.49%) — li_pc (2.75%) — li_oop, li_oop (2.57%) — q (2.45%)	25.08%
u2v10cd	li_oop (10.93%) — li_q (7.79%) — q (4.03%) — lo (2.66%) — li_pc (2.6%)	28.01%
u2v10cc	li_oop (11.0%) — li_q (7.85%) — q (3.97%) — li_pc (2.62%) — lo (2.62%)	28.06%
u2v10ad	li_oop (9.57%) — li_q (7.31%) — li_pc (2.65%) — li_oop, li_oop (2.54%) — q (2.35%)	24.42%
u2v10ac	li_oop (9.82%) — li_q (7.51%) — li_pc (2.74%) — li_oop, li_oop (2.58%) — li_oop, lo (2.38%)	25.03%
u2vccd	li_oop (10.87%) — li_q (7.7%) — q (3.92%) — li_pc (2.62%) — li_oop, lo (2.54%)	27.65%
u2vccc	li_oop (10.97%) — li_q (7.77%) — q (3.89%) — li_pc (2.65%) — li_oop, lo (2.57%)	27.85%
u2vcad	li_oop (9.49%) — li_q (7.19%) — li_pc (2.66%) — li_oop, li_oop (2.56%) — li_oop, lo (2.29%)	24.19%
u2vcac	li_oop (9.73%) — li_q (7.38%) — li_pc (2.75%) — li_oop, li_oop (2.61%) — li_oop, lo (2.35%)	24.82%
lti5cdb1	li_oop (18.29%) — li_q (7.01%) — li_oop, lo (4.41%) — oop (4.24%) — li_pc (3.2%)	37.15%
lti30cdb1	li_oop (17.17%) — li_q (7.06%) — li_oop, lo (3.9%) — oop (3.76%) — li_pc (3.01%)	34.9%
lti1cdb1	li_oop (14.11%) — li_q (7.25%) — li_oop, lo (3.32%) — li_pc (2.85%) — q (2.62%)	30.15%
lti14cdb1	li_oop (11.02%) — li_q (7.19%) — q (3.01%) — li_pc (2.62%) — li_oop, lo (2.57%)	26.41%
lti1adb1	li_oop (14.53%) — li_q (6.73%) — li_oop, lo (3.5%) — li_pc (2.9%) — li_oop, li_oop (2.75%)	30.41%
lti14adb1	li_oop (11.1%) — li_q (6.65%) — li_oop, li_oop (2.81%) — li_pc (2.76%) — li_oop, lo (2.64%)	25.96%
lti75adb1	li_oop (8.57%) — li_q (5.86%) — li_oop, li_oop (2.85%) — li_pc (2.26%) — li_oop, lo (2.03%)	21.57%
lti180adb1	li_oop (7.58%) — li_q (5.44%) — li_oop, li_oop (2.86%) — li_pc (2.0%) — li_oop, lo (1.8%)	19.68%

Table A3: Top 5 sequences for session approaches.

A4 User measures for td sessions regarding visited content

	∅Root Categories		∅Categories		∅Products		∅Queries		∅Logics	
	AM	SD	AM	SD	AM	SD	AM	SD	AM	SD
tdc	1.34	0.55	1.6	0.93	1.34	1.38	0.69	1.1	1.37	0.64
tdcm	1.34	0.55	1.6	0.93	1.34	1.38	0.69	1.1	1.37	0.64
tdp	1.34	0.55	1.6	0.93	1.33	1.37	0.68	1.08	1.36	0.64
tdpc	1.34	0.55	1.59	0.93	1.33	1.37	0.68	1.08	1.36	0.64
tdpcd	1.33	0.55	1.59	0.91	1.32	1.34	0.67	1.05	1.36	0.63
tdpd	1.34	0.55	1.59	0.92	1.33	1.35	0.68	1.06	1.36	0.63
tdpm	1.34	0.55	1.6	0.93	1.33	1.37	0.68	1.08	1.36	0.64
tdpr	1.34	0.55	1.6	0.93	1.33	1.37	0.68	1.08	1.36	0.64
tdr	1.34	0.55	1.6	0.94	1.34	1.39	0.69	1.1	1.37	0.64

Table A4: User measures for td sessions regarding visited content. Abbreviations: AM = Arithmetic Mean; SD = Standard Deviation.

A5 User measures for td sessions regarding time spent

	∅Time in session		∅Inter-Interactiontime		∅Interaction days	
	AM	SD	AM	SD	AM	SD
tdc	5.46	9.57	2.03	3.63	1.0	0.03
tdcm	5.46	10.91	2.03	6.53	1.0	0.03
tdp	5.35	9.31	1.99	3.54	1.0	0.03
tdpc	5.36	11.6	2.02	10.29	1.0	0.03
tdpcd	5.23	37.33	2.07	44.85	1.0	0.03
tdpd	5.2	8.87	2.02	3.74	1.0	0.03
tdpm	5.35	9.32	1.99	3.56	1.0	0.03
tdpr	5.35	9.38	2.0	3.61	1.0	0.03
tdr	5.46	9.54	2.02	3.55	1.0	0.03

Table A5: User measures for td sessions regarding time spent. Abbreviations: AM = Arithmetic Mean; SD = Standard Deviation.

A6 User measures for bm25ti sessions regarding system usage

	∅Sessions		∅Interactions		CV-R		B-R		Lead-ins	
	AM	SD	AM	SD	AM	SD	AM	SD	AM	SD
bm25ti5cc	5.84	19.86	3.11	2.75	60.25	109.97	38.74	33.91	1.28	0.99
bm25ti5cd	5.9	20.18	3.07	2.69	59.62	109.04	38.87	33.84	1.27	0.98
bm25ti30cc	5.19	17.37	3.65	3.5	74.15	135.95	33.31	33.78	1.47	1.15
bm25ti30cd	5.26	17.78	3.59	3.41	73.14	134.35	33.46	33.71	1.46	1.14
bm25ti1cc	4.42	13.67	4.16	4.18	85.06	154.49	28.06	32.56	1.68	1.42
bm25ti1cd	4.51	14.29	4.08	4.05	83.61	152.08	28.22	32.51	1.66	1.4
bm25ti1ac	3.97	9.46	4.33	4.28	88.19	156.75	26.18	31.6	1.73	1.43
bm25ti1ad	4.08	10.32	4.24	4.12	86.47	153.94	26.34	31.55	1.71	1.41
bm25ti14ac	3.15	5.32	4.97	5.46	100.68	177.65	22.66	30.36	2.01	2.06
bm25ti14ad	3.3	6.64	4.81	5.07	97.76	172.13	22.8	30.31	1.97	1.96
bm25ti75ac	2.82	4.01	5.28	6.03	106.6	187.23	21.07	29.64	2.16	2.35
bm25ti75ad	3.01	5.54	5.06	5.46	102.73	179.53	21.2	29.58	2.09	2.19
bm25ti180ac	2.73	3.63	5.37	6.17	108.26	189.43	20.58	29.39	2.2	2.42
bm25ti180ad	2.93	5.24	5.13	5.55	104.06	181.06	20.71	29.33	2.12	2.24

Table A6: User measures for bm25ti sessions regarding system usage. Abbreviations: CV-R = Conversion Rate; B-R = Bounce Rate; AM = Arithmetic Mean; SD = Standard Deviation.

A7 User measures for bm25ti sessions regarding visited content

	∅Root Categories		∅Categories		∅Products		∅Queries		∅Logics	
	AM	SD	AM	SD	AM	SD	AM	SD	AM	SD
bm25ti5cc	1.17	0.33	1.25	0.43	1.02	0.94	0.46	0.62	1.01	0.09
bm25ti5cd	1.17	0.32	1.24	0.41	1.01	0.93	0.46	0.61	1.01	0.08
bm25ti30cc	1.19	0.35	1.28	0.47	1.13	1.13	0.5	0.69	1.01	0.1
bm25ti30cd	1.19	0.35	1.27	0.45	1.12	1.11	0.49	0.67	1.01	0.09
bm25ti1cc	1.21	0.37	1.31	0.5	1.23	1.28	0.53	0.74	1.02	0.11
bm25ti1cd	1.2	0.36	1.3	0.48	1.21	1.26	0.52	0.71	1.02	0.1
bm25ti1ac	1.21	0.37	1.33	0.51	1.25	1.3	0.55	0.76	1.02	0.12
bm25ti1ad	1.21	0.37	1.32	0.49	1.24	1.27	0.54	0.73	1.02	0.1
bm25ti14ac	1.23	0.39	1.37	0.54	1.36	1.45	0.59	0.82	1.04	0.16
bm25ti14ad	1.23	0.38	1.35	0.51	1.34	1.4	0.58	0.77	1.03	0.15
bm25ti75ac	1.24	0.39	1.4	0.56	1.43	1.54	0.62	0.86	1.05	0.22
bm25ti75ad	1.24	0.39	1.38	0.53	1.39	1.46	0.6	0.79	1.05	0.21
bm25ti180ac	1.25	0.4	1.41	0.57	1.45	1.56	0.62	0.87	1.06	0.25
bm25ti180ad	1.24	0.39	1.39	0.53	1.41	1.47	0.6	0.79	1.05	0.24

Table A7: User measures for bm25ti sessions regarding visited content. Abbreviations: AM = Arithmetic Mean; SD = Standard Deviation.

A8 User measures for bm25ti sessions regarding time spent

	∅Time in session		∅Inter-Interactiontime		∅Interaction days	
	AM	SD	AM	SD	AM	SD
bm25ti5cc	1.48	10.41	1.13	11.07	1.0	0.01
bm25ti5cd	1.46	10.4	1.13	11.07	1.0	0.01
bm25ti30cc	3.77	11.9	1.98	11.06	1.0	0.02
bm25ti30cd	3.7	11.84	1.98	11.06	1.0	0.02
bm25ti1cc	70.93	232.99	32.18	111.64	1.05	0.19
bm25ti1cd	69.48	228.97	32.13	111.65	1.05	0.19
bm25ti1ac	77.91	243.88	33.38	112.65	1.05	0.2
bm25ti1ad	75.95	238.58	33.33	112.65	1.05	0.19
bm25ti14ac	1,113.01	3,286.21	362.92	1,211.14	1.23	0.72
bm25ti14ad	1,054.49	3,127.47	361.95	1,210.97	1.22	0.67
bm25ti75ac	4,216.47	12,405.08	1,421.57	5,401.83	1.32	0.93
bm25ti75ad	3,915.77	11,646.22	1,413.65	5,398.13	1.3	0.85
bm25ti180ac	7,154.38	22,292.71	2,509.54	10,664.05	1.35	0.98
bm25ti180ad	6,607.32	20,991.48	2,489.37	10,651.26	1.32	0.89

Table A8: User measures for bm25ti sessions regarding time spent. Abbreviations: AM = Arithmetic Mean; SD = Standard Deviation.

A9 User measures for combined u2v sessions regarding system usage

	∅Sessions		∅Interactions		CV-R		B-R		Lead-ins	
	AM	SD	AM	SD	AM	SD	AM	SD	AM	SD
u2v05ti5cc	5.82	19.84	3.14	2.8	60.65	110.69	38.57	33.97	1.29	0.99
u2v05ti5cd	5.83	19.95	3.13	2.78	60.51	110.4	38.6	33.96	1.28	0.99
u2v05ti30cc	5.17	17.4	3.7	3.58	74.76	137.19	33.15	33.81	1.48	1.16
u2v05ti30cd	5.19	17.55	3.68	3.54	74.49	136.61	33.19	33.8	1.48	1.16
u2v05ti1cc	4.41	13.84	4.21	4.29	85.83	156.29	27.92	32.58	1.7	1.43
u2v05ti1cd	4.44	14.08	4.19	4.23	85.41	155.32	27.96	32.57	1.69	1.43
u2v05ti14cc	3.9	12.7	4.69	5.21	95.32	174.06	24.76	31.6	1.92	1.98
u2v05ti14cd	3.94	12.99	4.65	5.08	94.67	172.26	24.82	31.59	1.91	1.93
u2v05ti1ac	3.93	9.46	4.4	4.41	89.31	158.85	25.89	31.53	1.75	1.44
u2v05ti1ad	3.96	9.75	4.37	4.33	88.84	157.74	25.93	31.53	1.75	1.44
u2v05ti14ac	3.11	5.33	5.06	5.66	102.11	180.89	22.41	30.28	2.04	2.12
u2v05ti14ad	3.16	5.81	5.0	5.45	101.18	178.2	22.46	30.27	2.02	2.05
u2v05ti75ac	2.79	4.02	5.37	6.27	108.13	191.48	20.89	29.57	2.18	2.41
u2v05ti75ad	2.85	4.56	5.29	5.95	106.72	187.03	20.93	29.56	2.16	2.31
u2v05ti180ac	2.7	3.64	5.46	6.41	109.76	193.77	20.43	29.34	2.22	2.47
u2v05ti180ad	2.77	4.2	5.37	6.06	108.18	188.87	20.48	29.32	2.2	2.37
u2v10ti5cc	5.79	19.68	3.15	2.81	60.91	110.9	38.32	33.94	1.29	0.99
u2v10ti5cd	5.82	19.86	3.13	2.78	60.55	110.31	38.38	33.91	1.28	0.99
u2v10ti30cc	5.14	17.21	3.71	3.6	75.12	137.49	32.86	33.75	1.49	1.16
u2v10ti30cd	5.17	17.45	3.68	3.53	74.53	136.41	32.93	33.72	1.48	1.16
u2v10ti1cc	4.37	13.57	4.23	4.31	86.29	156.66	27.6	32.47	1.7	1.43
u2v10ti1cd	4.42	13.95	4.19	4.21	85.44	155.01	27.68	32.45	1.69	1.42
u2v10ti14cc	3.86	12.41	4.72	5.22	95.87	174.5	24.43	31.46	1.92	1.95
u2v10ti14cd	3.92	12.85	4.65	5.05	94.65	171.76	24.51	31.44	1.91	1.92
u2v10ti1ac	3.92	9.35	4.41	4.42	89.52	159.06	25.73	31.5	1.75	1.44
u2v10ti1ad	3.98	9.86	4.36	4.3	88.53	157.13	25.81	31.47	1.74	1.43
u2v10ti14ac	3.1	5.24	5.07	5.66	102.34	181.1	22.24	30.22	2.04	2.09
u2v10ti14ad	3.19	6.03	4.97	5.37	100.59	177.05	22.3	30.19	2.01	2.03
u2v10ti75ac	2.78	3.93	5.38	6.27	108.34	191.43	20.7	29.5	2.18	2.39
u2v10ti75ad	2.89	4.84	5.25	5.83	105.92	185.39	20.76	29.47	2.14	2.28
u2v10ti180ac	2.69	3.55	5.47	6.41	109.98	193.73	20.24	29.26	2.22	2.45
u2v10ti180ad	2.8	4.5	5.32	5.93	107.31	187.08	20.3	29.23	2.18	2.33
u2vcti5cc	5.73	19.4	3.19	2.85	61.41	111.53	37.92	33.94	1.3	1.0
u2vcti5cd	5.75	19.56	3.17	2.82	61.1	110.99	37.98	33.92	1.3	1.0
u2vcti30cc	5.07	16.88	3.77	3.66	75.9	138.62	32.42	33.7	1.51	1.17
u2vcti30cd	5.1	17.1	3.73	3.59	75.38	137.58	32.49	33.67	1.5	1.17
u2vcti1cc	4.29	13.15	4.3	4.4	87.41	158.41	27.1	32.35	1.73	1.45
u2vcti1cd	4.34	13.51	4.26	4.3	86.63	156.74	27.18	32.33	1.71	1.44
u2vcti14cc	3.76	11.93	4.81	5.39	97.51	177.4	23.85	31.26	1.96	2.02
u2vcti14cd	3.82	12.36	4.74	5.18	96.32	174.4	23.93	31.25	1.94	1.95
u2vcti1ac	3.86	9.11	4.48	4.51	90.58	160.82	25.32	31.4	1.78	1.46
u2vcti1ad	3.91	9.57	4.43	4.39	89.68	158.89	25.39	31.38	1.76	1.44
u2vcti14ac	3.03	4.99	5.17	5.87	104.1	184.47	21.74	30.04	2.08	2.17
u2vcti14ad	3.11	5.75	5.07	5.53	102.37	179.96	21.81	30.02	2.05	2.07
u2vcti75ac	2.7	3.68	5.52	6.55	110.75	196.4	20.12	29.26	2.23	2.48
u2vcti75ad	2.8	4.56	5.37	6.04	108.2	189.3	20.19	29.24	2.19	2.34
u2vcti180ac	2.6	3.31	5.61	6.71	112.61	199.07	19.63	29.0	2.28	2.55
u2vcti180ad	2.71	4.23	5.45	6.15	109.77	191.22	19.69	28.97	2.23	2.39

Table A9: User measures for combined u2v sessions regarding system usage. Abbreviations: CV-R = Conversion Rate; B-R = Bounce Rate; AM = Arithmetic Mean; SD = Standard Deviation.

A10 User measures for combined u2v sessions regarding visited content

	⌀Root Categories		⌀Categories		⌀Products		⌀Queries		⌀Logics	
	AM	SD	AM	SD	AM	SD	AM	SD	AM	SD
u2v10ti5cc	1.17	0.33	1.26	0.46	1.03	0.95	0.46	0.63	1.0	0.04
u2v10ti5cd	1.17	0.33	1.26	0.44	1.02	0.95	0.46	0.62	1.0	0.04
u2v10ti30cc	1.19	0.35	1.3	0.51	1.15	1.15	0.5	0.7	1.0	0.05
u2v10ti30cd	1.19	0.35	1.29	0.49	1.14	1.14	0.5	0.68	1.0	0.04
u2v10ti1cc	1.21	0.37	1.33	0.54	1.24	1.31	0.54	0.75	1.0	0.05
u2v10ti1cd	1.21	0.37	1.32	0.52	1.23	1.29	0.53	0.73	1.0	0.05
u2v10ti14cc	1.22	0.38	1.36	0.56	1.33	1.44	0.57	0.79	1.01	0.1
u2v10ti14cd	1.22	0.38	1.35	0.54	1.32	1.41	0.56	0.76	1.01	0.1
u2v10ti1ac	1.21	0.37	1.35	0.55	1.27	1.32	0.56	0.78	1.01	0.06
u2v10ti1ad	1.21	0.37	1.34	0.53	1.26	1.3	0.55	0.76	1.01	0.05
u2v10ti14ac	1.23	0.39	1.39	0.58	1.38	1.49	0.6	0.84	1.02	0.11
u2v10ti14ad	1.23	0.38	1.38	0.55	1.36	1.44	0.59	0.8	1.01	0.11
u2v10ti75ac	1.24	0.39	1.42	0.6	1.44	1.57	0.63	0.88	1.02	0.18
u2v10ti75ad	1.24	0.39	1.41	0.57	1.42	1.51	0.61	0.82	1.02	0.17
u2v10ti180ac	1.24	0.4	1.43	0.61	1.46	1.59	0.63	0.89	1.03	0.21
u2v10ti180ad	1.24	0.39	1.42	0.57	1.44	1.52	0.62	0.83	1.03	0.21
u2v05ti5cc	1.17	0.32	1.26	0.46	1.03	0.95	0.46	0.64	1.0	0.02
u2v05ti5cd	1.17	0.32	1.26	0.45	1.02	0.95	0.46	0.63	1.0	0.02
u2v05ti30cc	1.19	0.35	1.3	0.51	1.14	1.15	0.5	0.7	1.0	0.02
u2v05ti30cd	1.19	0.35	1.29	0.49	1.14	1.14	0.5	0.69	1.0	0.02
u2v05ti1cc	1.2	0.37	1.33	0.54	1.24	1.3	0.54	0.75	1.0	0.02
u2v05ti1cd	1.2	0.36	1.32	0.52	1.23	1.29	0.53	0.74	1.0	0.02
u2v05ti14cc	1.22	0.38	1.36	0.57	1.32	1.44	0.57	0.79	1.01	0.09
u2v05ti14cd	1.22	0.38	1.35	0.54	1.31	1.41	0.56	0.77	1.01	0.09
u2v05ti1ac	1.21	0.37	1.34	0.55	1.27	1.32	0.56	0.78	1.0	0.04
u2v05ti1ad	1.21	0.37	1.34	0.53	1.26	1.3	0.55	0.77	1.0	0.04
u2v05ti14ac	1.23	0.38	1.38	0.58	1.38	1.48	0.6	0.84	1.01	0.1
u2v05ti14ad	1.22	0.38	1.37	0.56	1.37	1.45	0.59	0.81	1.01	0.1
u2v05ti75ac	1.24	0.39	1.41	0.6	1.44	1.56	0.62	0.88	1.02	0.17
u2v05ti75ad	1.23	0.39	1.4	0.57	1.42	1.52	0.61	0.84	1.02	0.17
u2v05ti180ac	1.24	0.39	1.42	0.61	1.46	1.58	0.63	0.89	1.02	0.2
u2v05ti180ad	1.24	0.39	1.41	0.58	1.44	1.53	0.62	0.84	1.02	0.2
u2vcti5cc	1.18	0.33	1.28	0.49	1.04	0.96	0.48	0.66	1.0	0.0
u2vcti5cd	1.18	0.33	1.27	0.47	1.03	0.95	0.47	0.65	1.0	0.0
u2vcti30cc	1.2	0.36	1.32	0.54	1.16	1.16	0.52	0.73	1.0	0.0
u2vcti30cd	1.2	0.36	1.31	0.52	1.15	1.15	0.51	0.71	1.0	0.0
u2vcti1cc	1.21	0.38	1.35	0.58	1.26	1.33	0.55	0.79	1.0	0.0
u2vcti1cd	1.21	0.37	1.34	0.55	1.25	1.31	0.55	0.76	1.0	0.0
u2vcti14cc	1.23	0.39	1.39	0.61	1.35	1.47	0.58	0.83	1.01	0.09
u2vcti14cd	1.22	0.38	1.38	0.57	1.34	1.44	0.58	0.8	1.01	0.08
u2vcti1ac	1.22	0.38	1.37	0.59	1.28	1.35	0.57	0.82	1.0	0.0
u2vcti1ad	1.22	0.38	1.36	0.56	1.27	1.32	0.56	0.79	1.0	0.0
u2vcti14ac	1.24	0.4	1.42	0.63	1.4	1.52	0.62	0.88	1.01	0.09
u2vcti14ad	1.23	0.39	1.4	0.59	1.38	1.47	0.61	0.84	1.01	0.09
u2vcti75ac	1.25	0.4	1.46	0.65	1.47	1.62	0.65	0.93	1.02	0.16
u2vcti75ad	1.24	0.4	1.44	0.61	1.45	1.54	0.63	0.86	1.01	0.16
u2vcti180ac	1.25	0.41	1.47	0.66	1.5	1.64	0.66	0.94	1.02	0.2
u2vcti180ad	1.25	0.4	1.45	0.62	1.47	1.56	0.64	0.87	1.02	0.2

Table A10: User measures for combined u2v sessions regarding visited content. Abbreviations: AM = Arithmetic Mean; SD = Standard Deviation.

A11 User measures for combined u2v sessions regarding time spent

	⌀Time in session		⌀Inter-Interactiontime		⌀Interaction days	
	AM	SD	AM	SD	AM	SD
u2v05ti5cc	1.5	10.41	1.13	11.07	1.0	0.01
u2v05ti5cd	1.49	10.41	1.13	11.07	1.0	0.01
u2v05ti30cc	3.82	11.94	1.98	11.06	1.0	0.02
u2v05ti30cd	3.8	11.92	1.98	11.06	1.0	0.02
u2v05ti1cc	71.36	234.5	32.15	111.58	1.05	0.19
u2v05ti1cd	70.96	233.43	32.14	111.58	1.05	0.19
u2v05ti14cc	865.78	2, 961.67	314.88	1, 158.99	1.19	0.63
u2v05ti14cd	855.4	2, 925.97	314.53	1, 158.95	1.18	0.62
u2v05ti1ac	78.96	246.3	33.43	112.66	1.05	0.2
u2v05ti1ad	78.46	244.93	33.42	112.66	1.05	0.2
u2v05ti14ac	1, 129.31	3, 333.9	361.62	1, 209.01	1.23	0.73
u2v05ti14ad	1, 109.42	3, 274.2	361.26	1, 208.86	1.23	0.71
u2v05ti75ac	4, 232.93	12, 571.01	1, 391.29	5, 336.68	1.33	0.95
u2v05ti75ad	4, 104.15	12, 183.21	1, 388.19	5, 335.24	1.32	0.92
u2v05ti180ac	7, 088.33	22, 353.65	2, 419.91	10, 436.03	1.35	1.0
u2v05ti180ad	6, 841.88	21, 668.5	2, 411.39	10, 430.17	1.34	0.96
u2v10ti5cc	1.51	10.42	1.13	11.07	1.0	0.01
u2v10ti5cd	1.49	10.41	1.13	11.06	1.0	0.01
u2v10ti30cc	3.84	11.96	1.98	11.06	1.0	0.02
u2v10ti30cd	3.8	11.92	1.98	11.05	1.0	0.02
u2v10ti1cc	71.77	235.05	32.14	111.51	1.05	0.19
u2v10ti1cd	70.98	233.06	32.11	111.51	1.05	0.19
u2v10ti14cc	869.58	2, 962.27	314.67	1, 157.29	1.19	0.63
u2v10ti14cd	852.8	2, 910.19	314.12	1, 157.28	1.18	0.62
u2v10ti1ac	78.98	246.34	33.36	112.54	1.05	0.2
u2v10ti1ad	77.95	243.72	33.33	112.53	1.05	0.2
u2v10ti14ac	1, 126.66	3, 324.87	360.69	1, 206.65	1.23	0.73
u2v10ti14ad	1, 092.89	3, 231.98	360.08	1, 206.45	1.23	0.7
u2v10ti75ac	4, 208.53	12, 463.48	1, 389.02	5, 323.06	1.33	0.95
u2v10ti75ad	4, 019.37	11, 949.84	1, 384.06	5, 320.3	1.31	0.9
u2v10ti180ac	7, 050.09	22, 155.62	2, 420.18	10, 415.75	1.35	1.0
u2v10ti180ad	6, 697.88	21, 258.63	2, 407.1	10, 406.04	1.34	0.94
u2vcti5cc	1.53	10.43	1.13	11.06	1.0	0.01
u2vcti5cd	1.52	10.42	1.13	11.06	1.0	0.01
u2vcti30cc	3.91	12.0	1.99	11.05	1.0	0.02
u2vcti30cd	3.87	11.96	1.98	11.05	1.0	0.02
u2vcti1cc	73.17	237.71	32.34	111.73	1.05	0.19
u2vcti1cd	72.41	235.74	32.31	111.73	1.05	0.19
u2vcti1ac	80.33	249.05	33.53	112.75	1.05	0.2
u2vcti1ad	79.36	246.48	33.5	112.74	1.05	0.2
u2vcti14cc	907.01	3, 045.89	322.65	1, 173.78	1.19	0.65
u2vcti14cd	888.63	2, 985.56	322.04	1, 173.74	1.19	0.63
u2vcti14ac	1, 173.28	3, 424.12	368.76	1, 223.22	1.24	0.75
u2vcti14ad	1, 137.41	3, 321.1	368.12	1, 223.0	1.23	0.72
u2vcti75ac	4, 506.58	13, 119.95	1, 448.43	5, 465.1	1.34	0.98
u2vcti75ad	4, 285.9	12, 503.82	1, 443.05	5, 462.28	1.33	0.93
u2vcti180ac	7, 626.21	23, 473.28	2, 542.37	10, 735.27	1.37	1.04
u2vcti180ad	7, 207.64	22, 384.72	2, 528.02	10, 725.85	1.35	0.97

Table A11: User measures for combined u2v sessions regarding time spent. Abbreviations: AM = Arithmetic Mean; SD = Standard Deviation.

A12 User measures for geometric sessions regarding system usage

	∅Sessions		∅Interactions		CV-R		B-R		Lead-ins	
	AM	SD	AM	SD	AM	SD	AM	SD	AM	SD
geomu24cd	3.78	9.95	4.78	4.9	95.15	168.52	23.09	30.89	1.87	1.5
geomu24cc	3.71	9.23	4.85	5.14	96.45	172.24	22.96	30.89	1.89	1.54
geomu14cd	3.16	7.91	5.45	6.32	108.13	192.95	19.55	29.31	2.15	2.14
geomu14cc	3.04	6.67	5.63	9.07	111.13	220.94	19.41	29.31	2.2	2.56
geomu24ad	3.58	8.13	4.87	4.98	96.88	169.99	22.28	30.39	1.9	1.51
geomu24ac	3.52	7.56	4.95	5.26	98.19	173.99	22.16	30.39	1.92	1.56
geomu14ad	2.79	4.68	5.69	6.74	112.4	198.46	18.54	28.65	2.23	2.28
geomu14ac	2.68	4.02	5.95	11.3	116.55	246.03	18.42	28.66	2.3	2.95
geomu75ad	2.45	3.46	6.18	7.99	121.74	218.09	16.63	27.55	2.44	2.8
geomu75ac	2.32	2.84	6.64	15.32	129.19	302.24	16.51	27.57	2.57	3.95

Table A12: User measures for geometric sessions regarding system usage. Abbreviations: CV-R = Conversion Rate; B-R = Bounce Rate; AM = Arithmetic Mean; SD = Standard Deviation.

A13 User measures for geometric sessions regarding time spent

	∅Time in session		∅Inter-Interactiontime		∅Interaction days	
	AM	SD	AM	SD	AM	SD
geomu24cd	74.05	238.48	29.97	106.42	1.05	0.19
geomu24cc	77.7	249.74	30.44	106.82	1.05	0.2
geomu14cd	904.81	3, 038.64	276.34	1, 039.61	1.21	0.7
geomu14cc	988.48	3, 387.25	281.95	1, 044.5	1.23	0.83
geomu24ad	78.38	245.7	30.71	107.01	1.05	0.2
geomu24ac	82.25	258.41	31.14	107.4	1.05	0.21
geomu14ad	1, 081.05	3, 304.64	306.28	1, 074.92	1.24	0.77
geomu14ac	1, 197.71	3, 806.39	311.76	1, 079.95	1.27	0.97
geomu75ad	3, 875.96	11, 947.43	1, 082.33	4, 429.35	1.37	1.09
geomu75ac	4, 443.74	13, 967.06	1, 106.15	4, 454.88	1.42	1.49

Table A13: User measures for geometric sessions regarding time spent. Abbreviations: AM = Arithmetic Mean; SD = Standard Deviation.

A14 System measures for all sessions approaches

	#Sessions	B-R	∅Root Categories	∅Categories	∅Products
visit_id	513,007,900	51.5%	1.18	1.27	0.89
bm25ti5cd	462,413,589	44.21%	1.16	1.22	0.95
bm25ti5cc	457,641,841	44.31%	1.16	1.22	0.96
u2v05ti5cd	457,028,558	44.39%	1.16	1.22	0.96
u2v05ti5cc	455,991,403	44.4%	1.16	1.22	0.96
u2v10ti5cd	455,774,632	44.03%	1.16	1.22	0.96
u2v10ti5cc	453,378,131	44.08%	1.16	1.23	0.96
u2vcti5cd	450,926,682	43.76%	1.16	1.23	0.97
u2vcti5cc	448,845,900	43.8%	1.16	1.23	0.97
lti5cdb1	428,209,378	42.1%	1.16	1.28	1.02
tf5	426,196,079	37.49%	1.22	1.35	1.06
bm25ti30cd	412,472,229	41.44%	1.17	1.25	1.0
bm25ti30cc	406,788,004	41.59%	1.17	1.25	1.01
u2v05ti30cd	406,679,566	41.74%	1.17	1.25	1.01
u2v10ti30cd	405,342,083	41.31%	1.17	1.26	1.02
u2v05ti30cc	405,332,467	41.75%	1.17	1.25	1.02
u2v10ti30cc	402,418,159	41.38%	1.17	1.26	1.02
u2vcti30cd	399,878,149	41.0%	1.18	1.27	1.03
u2vcti30cc	397,310,961	41.05%	1.18	1.27	1.03
ti5	388,179,295	38.63%	1.24	1.38	1.12
tf10	379,381,854	36.07%	1.24	1.39	1.14
lti30cdb1	373,299,068	39.07%	1.17	1.33	1.1
tf15	359,698,513	35.37%	1.26	1.42	1.17
ti10	358,125,465	36.8%	1.25	1.41	1.17
bm25ti1cd	353,801,526	37.06%	1.19	1.29	1.06
tf20	348,197,625	34.9%	1.26	1.43	1.2
u2v05ti1cd	347,716,468	37.5%	1.19	1.29	1.07
bm25ti1cc	346,643,630	37.28%	1.19	1.29	1.08
u2v10ti1cd	346,305,355	36.95%	1.19	1.3	1.08
u2v05ti1cc	345,822,886	37.52%	1.19	1.29	1.08
ti15	344,444,702	35.9%	1.26	1.43	1.2
u2v10ti1cc	342,511,906	37.04%	1.19	1.3	1.09
u2vcti1cd	339,887,635	36.55%	1.2	1.31	1.1
u2vcti1cc	336,459,169	36.63%	1.2	1.31	1.1
tf30	334,675,756	34.27%	1.27	1.46	1.23
ti26	329,331,338	34.83%	1.28	1.46	1.24
ti30	325,924,065	34.57%	1.28	1.46	1.24
tf45	323,595,977	33.64%	1.28	1.47	1.25
tdpcd	322,025,620	33.71%	1.28	1.47	1.25
tdpd	321,777,937	33.77%	1.28	1.47	1.25
tdpc	320,865,062	33.87%	1.28	1.48	1.25
tdpr	320,760,829	33.9%	1.28	1.48	1.25
tdp	320,740,004	33.93%	1.28	1.48	1.25
tdpm	320,732,793	33.92%	1.28	1.48	1.25
bm25ti1ad	319,821,010	33.11%	1.21	1.32	1.14
tdcm	319,460,083	33.86%	1.29	1.48	1.26
tdc	319,449,442	33.86%	1.29	1.48	1.26
tdr	319,385,497	33.9%	1.29	1.48	1.26
ti45	317,139,738	33.86%	1.29	1.48	1.27
tf60	316,804,935	33.2%	1.29	1.49	1.27
u2v10ti1ad	311,566,308	33.06%	1.2	1.33	1.16
bm25ti1ac	311,471,360	33.42%	1.21	1.32	1.16
ti60	311,459,298	33.36%	1.29	1.49	1.28
u2v05ti1ad	310,045,159	33.35%	1.2	1.32	1.16
bm25cd	308,865,935	34.55%	1.21	1.34	1.13
u2v05ti14cd	308,368,930	35.34%	1.2	1.32	1.13
tf90	308,159,712	32.53%	1.3	1.5	1.29
u2v05ti1ac	308,130,602	33.41%	1.2	1.32	1.16
u2v10ti1ac	307,315,533	33.22%	1.2	1.33	1.17
u2v10ti14cd	306,960,624	34.68%	1.21	1.34	1.14
u2vcti1ad	306,540,785	32.82%	1.21	1.34	1.18
u2v05ti14cc	305,770,681	35.38%	1.2	1.33	1.14
ti90	303,731,866	32.66%	1.3	1.51	1.3
lti1cdb1	303,701,559	34.05%	1.19	1.41	1.21
u2v05cd	302,896,160	35.23%	1.2	1.33	1.14
u2vcti1ac	302,831,132	32.95%	1.21	1.34	1.18
tf120	302,448,723	32.04%	1.3	1.51	1.31
u2v10ti14cc	302,146,443	34.83%	1.21	1.34	1.15
u2v10cd	301,526,828	34.54%	1.21	1.34	1.15
u2v05cc	300,045,032	35.27%	1.2	1.33	1.15
bm25cc	299,628,724	34.89%	1.21	1.34	1.15
u2vcti14cd	299,303,431	34.19%	1.21	1.36	1.16
ti120	298,319,596	32.16%	1.3	1.52	1.32
u2v10cc	296,737,824	34.62%	1.21	1.35	1.16
geomu24cd	296,104,692	32.52%	1.24	1.43	1.25

u2vcti14cc	294,754,777	34.31%	1.21	1.36	1.17
tf180	294,520,625	31.32%	1.31	1.53	1.33
u2vccd	293,231,413	33.99%	1.21	1.37	1.17
geomu24cc	290,945,430	32.61%	1.24	1.44	1.26
ti180	290,545,289	31.44%	1.31	1.54	1.34
u2vccc	288,285,363	34.13%	1.21	1.37	1.19
tf360	281,207,349	30.08%	1.32	1.56	1.37
geomu24ad	280,213,676	30.7%	1.25	1.45	1.29
ti360	277,612,404	30.16%	1.33	1.57	1.38
lti1adb1	276,414,965	30.97%	1.2	1.45	1.29
geomu24ac	275,590,269	30.82%	1.25	1.46	1.3
tf720	267,565,842	28.61%	1.34	1.6	1.42
tfd	267,188,092	28.8%	1.34	1.6	1.42
ti720	263,709,857	28.73%	1.34	1.6	1.42
bm25ti14ad	258,863,980	29.62%	1.23	1.4	1.26
u2v10ti14ad	249,713,150	29.8%	1.22	1.4	1.28
geomu14cd	247,540,067	29.67%	1.27	1.52	1.37
u2v05ti14ad	247,383,439	30.28%	1.22	1.38	1.28
lti14cdb1	246,895,650	30.5%	1.21	1.53	1.37
tf1440	246,739,006	26.33%	1.37	1.66	1.49
bm25ti14ac	246,474,512	30.31%	1.23	1.39	1.28
u2v05ti14ac	243,850,987	30.43%	1.22	1.38	1.28
u2vcti14ad	243,611,312	29.48%	1.23	1.42	1.31
u2v10ti14ac	242,926,971	30.18%	1.22	1.39	1.29
geomu14cc	238,492,729	29.92%	1.26	1.53	1.4
u2vcti14ac	237,279,638	29.81%	1.22	1.42	1.32
bm25ti75ad	235,749,939	27.63%	1.25	1.46	1.35
ti1440	235,395,534	26.83%	1.37	1.67	1.52
bm25ad	231,283,812	27.76%	1.24	1.47	1.37
lcdb1	229,523,765	28.96%	1.21	1.61	1.45
bm25ti180ad	229,260,341	26.92%	1.25	1.48	1.38
u2v10ti75ad	226,320,634	27.99%	1.24	1.45	1.37
u2v05ti75ad	223,642,592	28.58%	1.23	1.43	1.37
u2v10ad	221,684,899	28.22%	1.23	1.47	1.39
bm25ti75ac	220,970,579	28.5%	1.24	1.45	1.39
u2v10ti180ad	219,750,212	27.33%	1.24	1.47	1.41
u2vcti75ad	219,474,949	27.56%	1.24	1.48	1.41
u2v05ti75ac	218,928,488	28.8%	1.23	1.43	1.38
u2v05ad	218,918,746	28.87%	1.22	1.44	1.39
geomu14ad	218,814,074	27.29%	1.27	1.55	1.46
u2v10ti75ac	217,898,455	28.47%	1.23	1.45	1.4
u2v05ti180ad	217,009,417	27.96%	1.23	1.45	1.4
bm25ac	215,728,569	28.72%	1.24	1.46	1.41
u2vcad	214,602,352	27.77%	1.24	1.5	1.43
u2v05ac	213,752,978	29.12%	1.22	1.44	1.41
bm25ti180ac	213,749,268	27.83%	1.25	1.47	1.43
u2vcti180ad	212,709,792	26.86%	1.25	1.51	1.45
u2v10ac	212,641,685	28.77%	1.23	1.46	1.42
u2v05ti180ac	211,920,760	28.19%	1.23	1.45	1.42
u2vcti75ac	211,303,578	28.0%	1.24	1.48	1.44
u2v10ti180ac	210,812,680	27.84%	1.24	1.47	1.44
geomu14ac	210,212,581	27.7%	1.27	1.56	1.49
u2vcac	205,756,066	28.27%	1.23	1.5	1.47
u2vcti180ac	203,974,303	27.32%	1.24	1.51	1.48
lti14adb1	195,715,922	25.99%	1.21	1.66	1.59
geomu75ad	192,194,826	25.09%	1.3	1.65	1.6
geomu75ac	182,000,121	25.58%	1.29	1.67	1.66
lti75adb1	156,285,988	21.6%	1.23	1.92	1.93
ladb1	145,886,471	20.74%	1.23	2.03	2.06
lti180adb1	144,934,232	19.7%	1.24	2.03	2.07

Table A14: System measures for all sessions approaches. Ordered by the amounts of session in descending order. Abbreviations: CV-R = Conversion Rate; B-R = Bounce Rate.

A15 Overview of statistics per cluster

	1	2	3	4	5	6	7	8	9	10	Noise
ti30											
user_ids	6.83	5.59	5.59	5.51	4.5	3.68	3.57	3.41	2.91	2.9	5.76
sessions	1	1	1	2	1	2	2	1	1	1	25.16
interactions	3	4	2	1	5	1.5	2	6	7	8	9.27
ti180											
user_ids	7.5	6.34	6.26	5.02	4.77	3.86	3.4	3.26	3.23	3.21	5.02
sessions	1	1	1	1	2	1	5	1	2	1	21.85
interactions	3	2	4	5	1	6	3.25	7	1.5	8	13.99
tfd											
user_ids	7.84	6.74	6.57	5.26	4.36	4.06	3.42	3.34	3.29	3.02	4.86
sessions	1	1	1	1	2	1	1	1	5	2	19.16
interactions	3	2	4	5	1	6	7	8	3.44	1.5	13.47
u2vccc											
user_ids	7.39	6.72	6.24	4.87	4.38	3.66	3.63	3.12	2.97	2.94	6.34
sessions	1	1	1	1	2	1	5	2	2	1	18.49
interactions	3	2	4	5	1	6	3.35	1.5	2	7	11.93
u2vccd											
user_ids	7.37	6.72	6.21	4.83	4.38	3.62	3.14	2.99	2.89	2.62	5.14
sessions	1	1	1	1	2	1	2	2	1	5	23.06
interactions	3	2	4	5	1	6	1.5	2	7	3.86	11.24
u2vcac											
user_ids	7.39	6.72	6.24	4.87	4.38	3.66	3.58	3.4	3.14	2.94	2.48
sessions	1	1	1	1	2	1	2	2	5	1	12.01
interactions	3	2	4	5	1	6	1.5	2	4.71	7	22.92
u2vcad											
user_ids	7.37	6.72	6.21	4.83	4.38	3.62	3.61	3.42	2.89	2.74	4.01
sessions	1	1	1	1	2	1	2	2	1	2	11.31
interactions	3	2	4	5	1	6	1.5	2	7	2.5	16.76
ladb1											
user_ids	8.42	7.73	7.26	5.79	4.46	4.42	3.62	3.37	3.24	2.94	2.42
sessions	1	1	1	1	1	4	1	2	1	2	3.9
interactions	3	2	4	5	6	8.28	7	1	8	1.5	36.25
lcdb1											
user_ids	8.42	7.73	7.26	5.79	4.46	3.62	3.37	3.24	2.49	2.41	5.43
sessions	1	1	1	1	1	1	2	1	2	2	14.96
interactions	3	2	4	5	6	7	1	8	1.5	2	17.46
u2vcti30cc											
user_ids	6.62	5.49	4.48	4.43	4.33	4	3.48	2.74	2.51	2.25	8.17
sessions	2	1	1	1	2	2	1	2	1	6	24.71
interactions	1	3	2	4	1.5	2	5	2.5	6	2.02	6.76
u2vcti30cd											
user_ids	6.62	5.47	4.48	4.41	4.35	4.01	3.45	2.76	2.48	2.18	9.51
sessions	2	1	1	1	2	2	1	2	1	7	22.38
interactions	1	3	2	4	1.5	2	5	2.5	6	2.56	6.55
u2vcti1cc											
user_ids	6.56	5.74	5.42	5.36	4.21	3.66	3.44	3.09	2.7	2.49	7.16
sessions	1	1	1	2	1	2	2	1	6	2	20.38
interactions	3	2	4	1	5	1.5	2	6	2.95	2.5	10.62
u2vcti1cd											
user_ids	6.54	5.74	5.4	5.36	4.17	3.68	3.46	3.06	2.68	2.52	6.97
sessions	1	1	1	2	1	2	2	1	6	2	22.13
interactions	3	2	4	1	5	1.5	2	6	2.86	2.5	8.73
u2vcti1ac											
user_ids	6.56	5.74	5.42	5.36	4.21	4.05	3.75	3.63	3.09	2.85	6.1
sessions	1	1	1	2	1	2	2	5	1	2	19.39
interactions	3	2	4	1	5	1.5	2	3.19	6	2.5	13.14
u2vcti1ad											
user_ids	6.54	5.74	5.4	5.36	4.17	4.07	3.78	3.06	2.87	2.47	5.74
sessions	1	1	1	2	1	2	2	1	2	6	20.06
interactions	3	2	4	1	5	1.5	2	6	2.5	3.21	10.4
u2vcti14cc											
user_ids	7.19	6.47	6.03	4.7	4.63	3.49	3.26	3.09	2.8	2.66	5.38
sessions	1	1	1	1	2	1	2	2	1	5	21.09
interactions	3	2	4	5	1	6	1.5	2	7	3.82	12.83
u2vcti14cd											

user_ids	7.16	6.47	6	4.65	4.63	3.77	3.45	3.28	3.1	2.76	5.26
sessions	1	1	1	1	2	5	1	2	2	1	22.32
interactions	3	2	4	5	1	3.24	6	1.5	2	7	12.72
u2vcti14ac											
user_ids	7.19	6.47	6.03	4.7	4.63	3.68	3.49	3.46	3.22	2.8	4.32
sessions	1	1	1	1	2	2	1	2	5	1	14.83
interactions	3	2	4	5	1	1.5	6	2	3.99	7	16.51
u2vcti14ad											
user_ids	7.16	6.47	6	4.65	4.63	3.7	3.47	3.45	3.24	2.76	4.29
sessions	1	1	1	1	2	2	2	1	5	2	13.91
interactions	3	2	4	5	1	1.5	2	6	3.88	2.5	17.76
lti1cdb1											
user_ids	7.08	6.13	5.91	4.97	4.66	3.51	3.36	3.2	2.86	2.58	5.66
sessions	1	1	1	2	1	1	2	2	1	1	20.68
interactions	3	2	4	1	5	6	1.5	2	7	8	10.98
lti1adb1											
user_ids	7.09	6.13	5.91	4.97	4.66	3.67	3.51	3.48	3.38	2.86	4.88
sessions	1	1	1	2	1	2	1	2	5	1	18.43
interactions	3	2	4	1	5	1.5	6	2	3.38	7	13.1
lti180adb1											
user_ids	8.5	7.84	7.32	5.83	4.49	4.35	3.65	3.26	3.25	2.87	1.12
sessions	1	1	1	1	1	4	1	2	1	2	5.47
interactions	3	2	4	5	6	8.38	7	1	8	1.5	60.57
geomu24cc											
user_ids	7.34	6.38	6.1	4.82	4.72	3.67	3.27	3.18	3.05	2.92	5.41
sessions	1	1	1	1	2	1	2	2	1	1	21.28
interactions	3	2	4	5	1	6	1.5	2	7	8	12.31
geomu24cd											
user_ids	7.31	6.38	6.06	4.78	4.72	3.62	3.48	3.3	3.21	3.01	5.33
sessions	1	1	1	1	2	1	5	2	2	1	22.42
interactions	3	2	4	5	1	6	3.29	1.5	2	7	10.27
geomu24ac											
user_ids	7.34	6.38	5.95	4.82	4.7	3.67	3.49	3.34	3.05	2.92	6.28
sessions	1	1	1	1	2	1	2	2	1	1	14.9
interactions	3	2	4	5	1	6	1.5	2	7	8	15.08
geomu24ad											
user_ids	7.31	6.38	6.06	4.78	4.72	3.62	3.49	3.37	3.01	2.81	4.87
sessions	1	1	1	1	2	1	2	2	1	1	20.42
interactions	3	2	4	5	1	6	1.5	2	7	8	12.3

Table A15: Overview of statistics per cluster. The table shows statistics for the 10 largest clusters and the noise partition. Listed per session in the respective cluster are: the share of **user_ids** per cluster, average number of sessions and average number of interactions..

A16 Overview of statistics per cluster with additional features

	1	2	3	4	5	6	7	8	9	10	Noise
ti30											
u.	5.51	1.22	0.35	0.13	0.13	0.12	0.1	0.09	0.09	0.08	18.46
s.	2.0	3.0	4.0	5.0	9.0	11.0	10.0	8.0	12.0	7.0	9.98
int.	1.0	1.0	1.0	1.0	2.77	2.51	2.65	2.98	2.43	3.51	6.57
c.	1.0	1.0	1.0	1.0	1.22	1.09	1.2	1.38	1.08	1.57	1.81
t.	0.0	0.0	0.0	0.0	2.2	2.07	2.47	2.57	1.79	2.88	8.69
ti180											
u.	4.77	1.01	0.29	0.14	0.1	0.1	0.1	0.09	0.09	0.09	19.3
s.	2.0	3.0	4.0	17.0	19.0	11.0	18.0	5.0	7.0	21.0	8.45
int.	1.0	1.0	1.0	2.82	2.83	2.63	2.72	1.0	3.73	2.86	7.09
c.	1.0	1.0	1.0	1.18	1.17	1.09	1.16	1.0	1.57	1.2	1.89
t.	0.0	0.0	0.0	7.01	6.45	5.51	6.05	0.0	7.52	8.63	27.7
tfd											
u.	4.36	0.88	0.23	0.16	0.11	0.11	0.1	0.08	0.07	0.07	19.76
s.	2.0	3.0	4.0	15.0	16.0	18.0	17.0	19.0	20.0	5.0	7.44
int.	1.0	1.0	1.0	3.08	3.03	3.15	2.84	3.09	3.09	1.0	7.24
c.	1.0	1.0	1.0	1.2	1.2	1.23	1.21	1.25	1.22	1.0	1.91
t.	0.0	0.0	0.0	24.91	19.44	27.93	21.06	29.22	32.46	0.0	84.34
u2vccc											
u.	4.38	1.23	0.35	0.13	0.07	0.07	0.05	0.05	0.05	0.05	20.1
s.	2.0	3.0	4.0	5.0	1.0	12.0	1.0	6.0	13.0	2.0	7.32
int.	1.0	1.0	1.0	1.0	13.0	3.16	2.0	1.0	3.17	1.5	6.79
c.	1.0	1.0	1.0	1.0	2.0	1.17	1.0	1.0	1.15	1.0	1.53
t.	0.0	0.0	0.0	0.0	5.87	1, 812.24	0.1	0.0	986.25	0.09	11, 374.02
u2vccd											
u.	4.38	1.23	0.35	0.13	0.07	0.05	0.05	0.05	0.05	0.05	19.75
s.	2.0	3.0	4.0	5.0	1.0	1.0	6.0	2.0	2.0	2.0	7.68
int.	1.0	1.0	1.0	1.0	13.0	2.0	1.0	1.5	1.5	1.5	6.6
c.	1.0	1.0	1.0	1.0	2.0	1.0	1.0	1.0	1.0	1.0	1.5
t.	0.0	0.0	0.0	0.0	5.88	0.1	0.0	0.09	0.1	0.07	10, 942.61
u2vcac											
u.	4.38	0.76	0.17	0.12	0.12	0.1	0.07	0.05	0.05	0.05	20.84
s.	2.0	3.0	4.0	14.0	13.0	15.0	1.0	1.0	10.0	2.0	4.52
int.	1.0	1.0	1.0	4.81	4.34	4.89	13.0	2.0	3.93	1.5	8.39
c.	1.0	1.0	1.0	1.42	1.33	1.4	2.0	1.0	1.2	1.0	1.65
t.	0.0	0.0	0.0	22, 636.06	18, 642.73	25, 292.42	5.87	0.1	11, 241.7	0.09	24, 589.64
u2vcad											
u.	4.38	0.76	0.17	0.13	0.11	0.08	0.07	0.07	0.06	0.05	21.04
s.	2.0	3.0	4.0	13.0	15.0	14.0	8.0	1.0	10.0	1.0	4.94
int.	1.0	1.0	1.0	4.41	4.89	3.92	4.9	13.0	3.91	2.0	7.89
c.	1.0	1.0	1.0	1.35	1.42	1.31	1.5	2.0	1.2	1.0	1.6
t.	0.0	0.0	0.0	17, 809.83	21, 490.82	17, 458.05	14, 993.83	5.88	11, 541.73	0.1	22, 628.2
ladb1											
u.	3.37	0.38	0.27	0.19	0.12	0.11	0.11	0.1	0.07	0.07	21.05
s.	2.0	3.0	8.0	4.0	5.0	4.0	5.0	4.0	5.0	1.0	2.56
int.	1.0	1.0	16.49	9.82	7.92	10.22	5.79	11.91	6.63	13.0	12.68
c.	1.0	1.0	3.46	2.75	2.4	3.0	2.0	3.25	2.2	2.0	2.41
t.	0.0	0.0	123, 079.5	76, 982.97	67, 865.26	67, 864.36	38, 431.3	75, 443.98	30, 775.95	5.8	58, 070.2
lcdb1											
u.	3.37	0.82	0.2	0.09	0.09	0.08	0.08	0.07	0.07	0.07	20.76
s.	2.0	3.0	4.0	17.0	18.0	9.0	9.0	1.0	10.0	5.0	5.12
int.	1.0	1.0	1.0	3.77	4.09	3.78	3.09	13.0	3.21	1.0	8.78
c.	1.0	1.0	1.0	1.35	1.45	1.33	1.22	2.0	1.2	1.0	1.92
t.	0.0	0.0	0.0	3, 350.61	3, 991.06	5, 410.83	3, 253.74	5.8	4, 291.37	0.0	25, 539.34
u2vcti30cc											
u.	6.62	1.91	0.61	0.26	0.11	0.09	0.09	0.08	0.08	0.08	16.93
s.	2.0	3.0	4.0	5.0	6.0	11.0	2.0	2.0	2.0	10.0	13.04
int.	1.0	1.0	1.0	1.0	1.0	2.53	1.5	1.5	1.5	2.33	5.24
c.	1.0	1.0	1.0	1.0	1.0	1.18	1.0	1.0	1.0	1.2	1.44
t.	0.0	0.0	0.0	0.0	0.0	1.75	0.09	0.1	0.07	1.43	7.73
u2vcti30cd											
u.	6.62	1.91	0.61	0.26	0.11	0.09	0.09	0.08	0.08	0.08	16.78
s.	2.0	3.0	4.0	5.0	6.0	11.0	2.0	2.0	14.0	2.0	13.28
int.	1.0	1.0	1.0	1.0	1.0	2.53	1.5	1.5	2.28	1.5	5.17
c.	1.0	1.0	1.0	1.0	1.0	1.18	1.0	1.0	1.07	1.0	1.42
t.	0.0	0.0	0.0	0.0	0.0	1.76	0.09	0.1	1.56	0.07	7.73

u2vcti1cc											
u.	5.36	1.52	0.46	0.18	0.07	0.07	0.06	0.06	0.06	0.06	18.88
s.	2.0	3.0	4.0	5.0	6.0	1.0	2.0	2.0	2.0	2.0	10.25
int.	1.0	1.0	1.0	1.0	1.0	13.0	1.5	1.5	1.5	1.5	5.72
c.	1.0	1.0	1.0	1.0	1.0	2.0	1.0	1.0	1.0	1.0	1.45
t.	0.0	0.0	0.0	0.0	0.0	5.87	0.09	0.1	0.07	0.12	177.3
u2vcti1cd											
u.	5.36	1.52	0.46	0.18	0.07	0.07	0.06	0.06	0.06	0.06	19.05
s.	2.0	3.0	4.0	5.0	6.0	1.0	2.0	2.0	2.0	2.0	10.36
int.	1.0	1.0	1.0	1.0	1.0	13.0	1.5	1.5	1.5	1.5	5.57
c.	1.0	1.0	1.0	1.0	1.0	2.0	1.0	1.0	1.0	1.0	1.43
t.	0.0	0.0	0.0	0.0	0.0	5.88	0.09	0.1	0.07	0.12	172.73
u2vcti1ac											
u.	5.36	1.13	0.31	0.1	0.07	0.07	0.06	0.06	0.06	0.06	18.72
s.	2.0	3.0	4.0	5.0	1.0	2.0	2.0	2.0	2.0	2.0	8.93
int.	1.0	1.0	1.0	1.0	13.0	1.5	1.5	1.5	1.5	1.5	6.16
c.	1.0	1.0	1.0	1.0	2.0	1.0	1.0	1.0	1.0	1.0	1.49
t.	0.0	0.0	0.0	0.0	5.87	0.09	0.1	0.07	0.12	0.17	201.32
u2vcti1ad											
u.	5.36	1.13	0.31	0.1	0.07	0.07	0.06	0.06	0.06	0.06	19.02
s.	2.0	3.0	4.0	5.0	1.0	2.0	2.0	2.0	2.0	2.0	9.03
int.	1.0	1.0	1.0	1.0	13.0	1.5	1.5	1.5	1.5	1.5	6.0
c.	1.0	1.0	1.0	1.0	2.0	1.0	1.0	1.0	1.0	1.0	1.47
t.	0.0	0.0	0.0	0.0	5.88	0.09	0.1	0.07	0.12	0.17	194.65
u2vcti14cc											
u.	4.63	1.3	0.37	0.14	0.07	0.06	0.05	0.05	0.05	0.05	20.67
s.	2.0	3.0	4.0	5.0	1.0	6.0	1.0	2.0	2.0	2.0	8.27
int.	1.0	1.0	1.0	1.0	13.0	1.0	2.0	1.5	1.5	1.5	6.54
c.	1.0	1.0	1.0	1.0	2.0	1.0	1.0	1.0	1.0	1.0	1.5
t.	0.0	0.0	0.0	0.0	5.87	0.0	0.1	0.1	0.09	0.07	2,134.12
u2vcti14cd											
u.	4.63	1.3	0.37	0.14	0.07	0.06	0.05	0.05	0.05	0.05	20.59
s.	2.0	3.0	4.0	5.0	1.0	6.0	1.0	2.0	2.0	13.0	8.41
int.	1.0	1.0	1.0	1.0	13.0	1.0	2.0	1.5	1.5	2.95	6.36
c.	1.0	1.0	1.0	1.0	2.0	1.0	1.0	1.0	1.0	1.15	1.48
t.	0.0	0.0	0.0	0.0	5.88	0.0	0.1	0.1	0.09	549.13	2,065.99
u2vcti14ac											
u.	4.63	0.87	0.22	0.13	0.13	0.08	0.08	0.07	0.07	0.07	19.65
s.	2.0	3.0	4.0	15.0	16.0	18.0	17.0	10.0	1.0	19.0	5.79
int.	1.0	1.0	1.0	3.43	3.67	3.49	3.46	3.33	13.0	3.68	7.74
c.	1.0	1.0	1.0	1.17	1.21	1.19	1.17	1.2	2.0	1.21	1.58
t.	0.0	0.0	0.0	1,460.15	1,436.81	1,637.33	1,465.54	1,307.09	5.87	1,653.58	3,086.45
u2vcti14ad											
u.	4.63	0.87	0.22	0.1	0.08	0.07	0.07	0.07	0.06	0.06	19.72
s.	2.0	3.0	4.0	16.0	10.0	17.0	18.0	1.0	5.0	19.0	6.16
int.	1.0	1.0	1.0	3.4	3.41	3.36	3.35	13.0	1.0	3.55	7.4
c.	1.0	1.0	1.0	1.16	1.2	1.16	1.16	2.0	1.0	1.18	1.55
t.	0.0	0.0	0.0	1,308.55	1,408.44	1,276.06	1,466.28	5.88	0.0	1,396.12	2,939.1
lti1cdb1											
u.	4.97	1.29	0.36	0.14	0.07	0.06	0.06	0.06	0.05	0.05	19.88
s.	2.0	3.0	4.0	5.0	1.0	20.0	2.0	2.0	2.0	1.0	8.76
int.	1.0	1.0	1.0	1.0	13.0	2.72	1.5	1.5	1.5	2.0	6.25
c.	1.0	1.0	1.0	1.0	2.0	1.13	1.0	1.0	1.0	1.0	1.6
t.	0.0	0.0	0.0	0.0	5.8	42.86	0.09	0.1	0.07	0.1	201.56
lti1adb1											
u.	4.97	0.99	0.26	0.11	0.09	0.08	0.07	0.06	0.06	0.06	18.87
s.	2.0	3.0	4.0	18.0	19.0	5.0	1.0	20.0	2.0	2.0	7.84
int.	1.0	1.0	1.0	2.98	3.02	1.0	13.0	3.0	1.5	1.5	6.75
c.	1.0	1.0	1.0	1.16	1.18	1.0	2.0	1.14	1.0	1.0	1.67
t.	0.0	0.0	0.0	66.43	74.53	0.0	5.8	68.67	0.09	0.1	234.16
lti180adb1											
u.	3.26	0.36	0.32	0.18	0.18	0.15	0.13	0.11	0.09	0.07	20.3
s.	2.0	3.0	8.0	5.0	5.0	5.0	4.0	5.0	9.0	1.0	2.53
int.	1.0	1.0	15.19	4.83	5.48	6.35	10.75	7.02	10.41	13.0	13.25
c.	1.0	1.0	3.23	1.6	1.8	2.0	3.0	2.2	2.65	2.0	2.46
t.	0.0	0.0	102,574.08	30,945.72	40,298.02	41,868.93	66,324.84	39,970.89	90,005.35	5.8	49,316.45
geomu24cc											
u.	4.72	1.13	0.32	0.11	0.07	0.06	0.06	0.06	0.05	0.05	19.49
s.	2.0	3.0	4.0	5.0	20.0	2.0	21.0	2.0	1.0	1.0	8.24
int.	1.0	1.0	1.0	1.0	2.82	1.5	2.72	1.5	2.0	2.0	6.64
c.	1.0	1.0	1.0	1.0	1.14	1.0	1.12	1.0	2.0	1.0	1.67
t.	0.0	0.0	0.0	0.0	47.26	0.09	33.67	0.1	0.1	0.1	191.55
geomu24cd											

u.	4.72	1.13	0.32	0.11	0.06	0.06	0.05	0.05	0.05	0.05	19.94
s.	2.0	3.0	4.0	5.0	2.0	2.0	1.0	1.0	2.0	2.0	8.52
int.	1.0	1.0	1.0	1.0	1.5	1.5	2.0	2.0	1.5	1.5	6.36
c.	1.0	1.0	1.0	1.0	1.0	1.0	2.0	1.0	1.0	1.0	1.63
t.	0.0	0.0	0.0	0.0	0.09	0.1	0.1	0.1	0.07	0.0	177.41
geomu24ac											
u.	4.72	0.94	0.26	0.08	0.08	0.06	0.06	0.06	0.05	0.05	19.24
s.	2.0	3.0	4.0	5.0	20.0	2.0	22.0	2.0	1.0	1.0	7.67
int.	1.0	1.0	1.0	1.0	2.95	1.5	2.94	1.5	2.0	2.0	6.92
c.	1.0	1.0	1.0	1.0	1.15	1.0	1.15	1.0	2.0	1.0	1.7
t.	0.0	0.0	0.0	0.0	59.93	0.09	56.51	0.1	0.1	0.1	206.25
geomu24ad											
u.	4.72	0.94	0.26	0.08	0.07	0.06	0.06	0.05	0.05	0.05	20.24
s.	2.0	3.0	4.0	5.0	20.0	2.0	2.0	1.0	1.0	2.0	7.64
int.	1.0	1.0	1.0	1.0	2.92	1.5	1.5	2.0	2.0	1.5	6.65
c.	1.0	1.0	1.0	1.0	1.15	1.0	1.0	1.0	2.0	1.0	1.67
t.	0.0	0.0	0.0	0.0	48.96	0.09	0.1	0.1	0.1	0.07	184.42

Table A16: Overview of statistics per cluster with additional features. The table shows statistics for the ten largest clusters and the noise partition. The share of user_ids per cluster as well as the average sessions and the averaged number of average interactions per session in the respective cluster. Abbreviations: u. = user_ids; s. = sessions; int. = interactions; c. = categories; t. = time on site.