# Using contextual information to understand searching and browsing behavior

# Using Contextual Information to Understand Searching and Browsing Behavior

# Using Contextual Information to Understand Searching and Browsing Behavior

PROEFSCHRIFT

ter verkrijging van de graad van doctor aan de Technische Universiteit Eindhoven, op gezag van de Rector Magnificus prof.dr.ir. F.P.T. Baaijens, voor een commissie aangewezen door het College voor Promoties, in het openbaar te verdedigen op maandag 13 juni 2016 om 14:00 uur

door

Yulia Kiseleva

geboren te Kaliningrad, Rusland

Dit proefschrift is goedgekeurd door de promotoren en de samenstelling van de promotiecommissie is als volgt:

| | |
|---|---|
| voorzitter: | prof.dr. J. de Vlieg |
| promotor: | prof.dr. P.M.E. De Bra |
| copromotor(en): | dr. M. Pechenizkiy |
| leden: | prof.dr. A. Tuzhilin (New York University) |
| | prof.dr.ir. A. P. de Vries (Radboud Universiteit Nijmegen) |
| | prof.dr. C.L.A. Clarke (University of Waterloo) |
| | prof.dr.ir. U. Kaymak |
| | prof.dr.ir. W.M.P. van der Aalst |

*Het onderzoek dat in dit proefschrift wordt beschreven is uitgevoerd in overeenstemming met de TU/e Gedragscode Wetenschapsbeoefening.*

# Using Contextual Information to Understand Searching and Browsing Behavior

**Julia Kiseleva**

SIKS Dissertation Series No. 2016-25
The research reported in this dissertation has been carried out under the auspices of SIKS, the Dutch Research School for Information and Knowledge Systems.


This research was supported by the Dutch Technology Foundation STW (project CAPA STW 11736), which is part of the Netherlands Organization for Scientific Research (NWO).



Cover by Agustina Huarte and Vadim Denisov.
Printed by ipskampprinting.nl

A catalog record is available at the TU/e's Library (`http://repository.tue.nl/`).
ISBN: 978-90-386-4077-8

# Contents

# List of Figures

# List of Tables

# Acknowledgments

> What is success? I think it is a mixture of having a flair for the thing that you are doing; knowing that it is not enough, that you have got to have hard work and a certain sense of purpose.
>
> Margaret Thatcher

My journey towards this dissertation started in 2007 when I finished my master degree at the Saint-Petersburg State University and had no idea how can I use all the "luggage of knowledge" acquired at university. I was fortunate to meet many great people who contributed to the person and the researcher I am today.

Back in 2007, I started to work with Igor Nekrestyanov and Marina Nekrestyanova. Igor and Marina, you were excellent mentors for me showing what is a real passion for research. You taught me how important is to ask right questions, seek for answers, and be skeptical about the results. I was very lucky to meet Ilya Segalovich on my way, who was a unique person because he could inspire and encourage your ideas and always could give you directions. One of the greatest lesson I learnt from Ilya is to always be open to new ideas and listen to others. It was a great time and a great learning experience to work in Hewlett Packard Labs and later Yandex. This industrial experience was extremely influential to my reasoning about research and my desire always bring practical value through my work.

Eugene Agichtein, who was my mentor during my stay in Emory University, and contributed a lot to the way I formulate and approach research problems now. Eugene, thanks for teaching me number of important lessons and introducing me to the IR community.

I am very grateful to my promotors, Paul De Bra and Mykola Pechenizkiy, for accepting me for this PhD position in March 2012 and giving me freedom to choose interesting problems to work on. During my first years, I also collaborated a lot with Hoang Thanh Lam and Toon Calders. Lam and Toon, thank you for our great discussions and teaching me a lot about academic life.

Graduate school is not just about producing many papers (which hopefully have great scientific impact) but it is a great time to meet new people, to explore the world (well, if your papers are fortunate to get through the 15-20% acceptance rate), but also to extend your background in many possible ways. I was very, very lucky to meet great people who contributed to this thesis either working together or giving me inspiration.

It was great and lucky chance to be invited to co-organize the TREC Contextual Suggestion Track where I met my great colleagues: Jaap, Charlie, Adriel and Hadi. Especially, I am in debt to Jaap. Jaap, I could not be more thankful

for your continuous inspiration and your support to continue my line of research, especially when it was not that easy, and to pursue my dreams no matter what.

During my graduate studies I was fortunate to do three internships at Microsoft Bing, Booking.com and Microsoft Research which have been resulted in a number of publications. This experience contributed significantly to my academic and production development.

Riccardo, Eric and Roland were my mentors during my stay in Bing. Thank you for giving me freedom and the same time always willing to help if I needed it. I hope that we will have a chance to work together in the future, and I hope to continue "exceeding your expectations."

My internship in Booking.com was truly exceptional as I was exposed to real-time online experimentation. Nothing is more exciting than see in real-time how your methods impact the experience of many users online. Thank you Melanie, Lucas and Mats for our fruitful discussions and productive collaborations. Lucas, thank you for criticizing my ideas so I can come up with better augmentations and reasoning. I also want to express my gratitude to Alexander Tuzhilin for guiding me during my stay in Booking.com.

My stay in Microsoft Research was unforgettable not just because of excellent opportunities for research but all the beauty of Washington area (at least in summer) and all the activities organized for interns. Ahmed, thank for being one of the greatest mentors I ever had. You have a rare quality to listen and trust the ideas of others. And the same time, if something goes wrong you are always willing to give a hand.

I am very thankful to my mentors at the SIGIR 2015 doctoral consortium: Diane Kelly and Mark Sanderson. Our discussion makes me finally realized how much work was already done and that I am ready to finish my PhD. Thank you for giving me clear advice on how to coherently organize all my work.

I am grateful to all my collaborators and co-authors around the world. This dissertation would not be what it is today without them: Ahmed Hassan Awadallah, Kyle Williams, Aidan C. Crook, Imed Zitouni, Tasos Anastasakos, Madian Khabsa, Jiepu Jiang, Riccardo Brigo, Eric Crestan, Roland Dittel, Vadim Nikulin, Nikita Makarov, Adriel Dean-Hall, Charles L.A. Clarke, Jaap Kamps, Hadi Hashemi, Lucas Bernardi, Melanie J.I. Müller, Alexander Tuzhilin, Chad Davis, Ivan Kovacek, Mats Stafseng Einarsen, Djoerd Hiemstra, Nikita Spirin, Mikhail Kuznetsov, Yaroslav Spirin, Pavel Izhutov, Alejandro Montes García, Mykola Pechenizkiy, Paul De Bra, Hoang Thanh Lam, Toon Calders, Yongming Luo.

I would like to extend my sincere gratitude to the members of my reading committee, for reviewing this dissertation and for your insightful comments: Alexander Tuzhilin from New York University, Arjen de Vries from Radboud University Nijmegen, Charlie Clarke from the University of Waterloo, Uzay Kaymak from Eindhoven University of Technology, and Wil van der Aalst from Eindhoven University of Technology.

Today, I am happy to be part of a great research community. It was wonderful to meet all these great people again and again wherever there was a conference. Just to mention two of the new friends I made: Grace, thank you for your advice,

which is very valuable to me, but most importantly thanks for being such a great researcher—the example for many including me. Maarten, the discussions we had mostly during conferences inspired me to see the opportunities and not the problems. Hopefully, we will have a chance to work together in future.

Thanks for all my colleagues in IS group for creating such a friendly working atmosphere. I miss our game nights a lot. Maybe the new generation of PhD students should think about renewing this tradition. I want to thank our great secretaries Ine and Riet who always took care about all bureaucratic burdens. And a special thanks to Riet for proofreading my dissertation.

Special gratitude goes to my family for their unconditional support. Дорогие мои родители! Папа и Мама! Спасибо Вам большое за вашу безграничную веру в меня и поддержку! Вы на самом деле удивительные родители, вы никогда не говорили мне что и как нужно делать. Вы всегда доверяли мне и с уважением относились к моим решениям и в то же время я знаю, что вы меня поддержите несмотря ни на что. Дорогие Людмила Васильевна и Владимир Васильевич, спасибо большое за вашу поддержку все эти годы— это очень важно для меня! Спасибо за ваш неподдельный интерес к моей работе.

Vadim, thank you for so many things but if I list all of them this thesis would be too long. Most importantly, I want to thank you for your help and support during these years, when nobody else could help you still managed! You know when you say *'Julia, you can do anything!'* I believe I can.

Finally, thanks to all my friends which made last four years very enjoyable: Dong; Natasha and Robert; Pedro Antonio; the 'cool' office: Elham, Dennis and Joos; Maxim; Ana-Maria; Alejandro; Lam; Riccardo; Rolland; Eric; Jonas and Anna; Pavels; Maya and Ilya; Jorge and Conny; Melanie; Lucas; Vova and Angelina; Katya and Slava; Misha and Katya; Lena and Stas; Sasha; Ivan and Natasha; and many, many others.

April, 2016 — Julia Kiseleva
In the air between Montreal and Amsterdam — (Юля Киселёва)

# 1 Introduction

> Knowledge must come through action you can have no test which is not fanciful, save by trial.
>
> Sophocles

Web analytics aims at understanding behavioral patterns of users of various web-based services in order to improve their experience. A better understanding of user interactions with a web service allows us to model user behavior and to predict future user actions. The ability to predict user preferences helps to serve suitable content, e. g. adequate advertisements, fitting recommendations, etc. Ultimately, the goal is to increase user satisfaction and engagement with the provided information.

The main source of data for those web analytics predictive tasks are the traces of user interaction behavior, which are inherently sensitive to context. Contextual information can be defined as a collection of external factors influencing user behavior, e. g. location, season, access device, weather, user situation. For instance, the number of hotel bookings increases during the holiday period. User interaction with search result pages (SERPs) on mobile devices is different from that on the desktop, as illustrated in Figure 1.1, due to the difference in the screen size and as a consequence different behavioral patterns. In short, user behavior can vary dramatically depending on the context. Thus complementing the web analytics modeling and prediction techniques with context management mechanisms hold the potential to make them customized and more accurate. The importance of contextual information has been recognised by researchers and practitioners across many disciplines, including recommendation systems, information retrieval, ubiquitous and mobile computing, behavioral targeting and marketing.

In this dissertation, we focus on the *contextual customization* of web services to user types, or personas, rather than *personalization* to specific individuals. Personalization is not always possible because not enough data is collected or

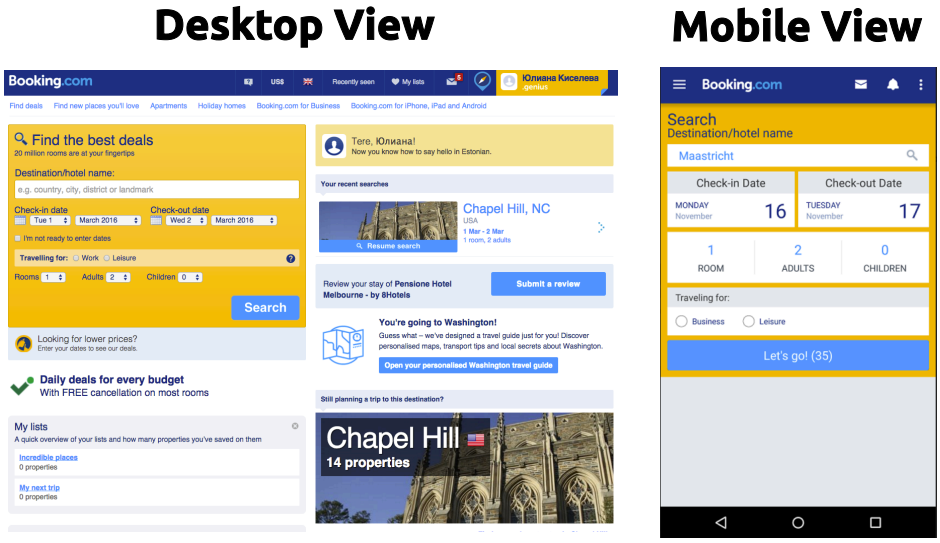



Figure 1.1: Illustration of Desktop (left) and Mobile (right) Views of Booking.com search page.

due to continuous changes of individual preferences [29]. In our research, we distinguish between two main types of user behavior:

- *Searching*—in case a web service needs to respond to a user request, such as a user issuing a query to a search engine, using contextual information can help to give more satisfying answers.

- *Browsing*—in case a web service needs to infer user preferences from user interaction behavior, such as a user clicking on recommended items, using contextual information can help to better predict the user's intent in the current situation.

Modern web search still relies on the query-response paradigm, which is characterized by a sharp contrast between the richness of data in the index, and the relative poverty of information in the query, usually expressed in a few keywords to express a complex need. This is particularly true in online search services, where the same query may be observed from many users, with considerable variations in their search intents. Contextual information is the obvious route to try to restore the balance, and behavioral data related to user's searching and browsing activities provides new opportunities to model contextual aspects of user needs. In this dissertation we investigate the principles that allow effective utilization of contextual information in various web applications, and translate our insights into new approaches for fast and accurate understanding and predicting user behavior.

The main research question that motivates the research presented in this dissertation is:

> *How to discover, model and utilize contextual information in order to understand and improve users' searching and browsing behavior on the web?*

## 1.1 What is (not) Context?

We will first give an overview of the use of context in an information access setting: what is contextual information? what web applications benefit of the use of contextual information? and what are the principled ways to integrate contextual information into the web analytics process.

Many interpretations of the notion of context have emerged in various fields of research like psychology, philosophy, and computer science [36]. Applied to web analytics, a context is presented as additional (situational) information about a user, e. g. geographical location [1, 110], current date [42, 128, 212], season [21, 209], weather [195], emotional status [40], or interaction pattern [11, 77, 97, 124]. Dey et al. [64] consider context even broader: *any information that can characterize and is relevant to the interaction between a user and an application*. In this dissertation we distinguish two types of context:

- *explicit context*—where we assume that contextual information is given by a domain expert and easily available with the data, e. g. an expert claims geographical location is context for this web service and it can be derived from the attendant web logs;
- *implicit context*—where we assume that contextual information is not directly known and available and we need to apply some discovery techniques to infer it [253, 255].

The summarized timeline of the notable examples of context modeling is presented in Table 1.1.[1]

We distinguish three types of web applications, where users perform searching and browsing behavior. We consider the context discovery and context modeling techniques typical for behavioral targeting, for recommender systems, and for information retrieval.

### 1.1.1 Context in Behavioral Targeting

Behavioral targeting is an active area of research [13, 18, 48, 83, 167, 169]. The goal of behavioral targeting is to build user profiles of interests fast while users are interacting with the system, e. g. clicking on ads, sponsored links, or searching for particular products. The history of user interactions is a useful source

[1] More detailed background for context discovery and modeling techniques is presented in Section 2.2.1

Table 1.1: Notable examples of context modeling techniques

| Context | Year |
|---|---|
| Location [17, 110, 238] | 1992 |
| Emotional status [40] | 1997 |
| Taxonomy of explicit context [126, 203] | 1999 |
| Predictive features versus contextual [231] | 2002 |
| Conceptual models as context [102, 103] | 2004 |
| History of previous interaction [180] | 2008 |
| Independence of predicted class [253] | 2011 |
| Two level prediction model [255] | 2012 |
| Focus on automatic context discovery | 2013– |

to predict user interests and build short–and long–term user profiles. The process of building user profiles ($P$) based on user interaction ($i_1, \ldots, i_n$) can be formalized as follows: $f : i_1, \ldots, i_n \rightarrow P$. One of the directions of behavioral targeting is the methods for online advertising where we need to understand user interests and to predict user preferences to serve suitable content ($A$), e. g. advertisements. The problem of serving suited content to users is formalized as the following prediction function: $H : P \times A \rightarrow [0, 1]$. The ads publishers have up-to-date contextual knowledge about their audiences such as demographic information [83] and a history of user behavior [13]. Some targeting methods are designed to match users with appropriate ads taking contextual information ($C$) into account [169]. Then, the behavioral targeting problem can be formalized as follows: $H : P \times A \times C \rightarrow [0, 1]$. The context of user behavior can consist of the following explicit factors: the page the user is currently visiting, time, and user historical behavior online [46]. Three types of targeting methods are popular in the advertising industry: property, user segment, and behavioral targeting. Property targeting refers to placing ads on specific web pages where interested users will appear, such as showing online brokerage ads on financial related pages. Although this reaches users who visit these finance pages, it may miss users who do not visit web financial resources but might be interested in finances. User segment targeting can focus on specific groups, e. g. based on gender and age of a user etc. However, it is also possible to target broad groups, e. g. women under 25 years old with high income. Behavioral targeting approaches use more contextual information to infer more specific user characteristics, e. g. history of user interaction. Whereas property targeting targets pages, and user segment targeting targets generic groups, behavioral targeting tackle more specific groups or even individuals. Behavioral targeting methods benefit from adopting contextual information.

### 1.1.2 Context in Recommender Systems

In classical formulations of recommender systems, the recommendation problem relies on ratings ($R$) as a mechanism of capturing user ($U$) preferences for different items ($I$). The problem of estimating unknown ratings is formalized as follows: $F : U \times I \rightarrow R$. The predicted ratings can be used for ranking of recommendations. The radical departure from the classical, two-dimensional recommender systems is context-aware recommender systems [3, 4], which attract increasing attention in academic work [91, 92, 208]. Rating prediction in context-aware recommender system relies primarily on the information of *how* (which rating, e. g. a user giving '3' of '5' stars to an item) and *who* (which user, e. g. age, gender, mood, or nationality [221]) rated *what* (which item, e. g. movie, news article, or hotel). This additional information is context [189]. The general formulation of context-aware recommender system rating prediction takes into account the context dimension $C$ as follows [4]: $F : U \times I \times C \rightarrow R$. For example, a temporal context has emerged for travel recommenders where suggestions about vacation in the winter are very different compared to summer [21]. The purchase intent of a customer is considered as implicit contextual information that needs to be predicted [6]. Then it can be used to select the suitable model. Therefore, the context-aware recommenders utilize the information about user situation to make better predictions.

### 1.1.3 Context in Information Retrieval

The ultimate goal of information retrieval is to develop the technology needed to provide access to data collections. Context is of particular importance in web search, where a heterogeneous body of information is served to a highly heterogeneous set of users. Users issue a query $Q$ and a search engine returns SERP that is a ranked list of URLs retrieved from the indexed collection of web pages ($D$): $SERP{=}(url_1, \ldots, url_i, \ldots, url_n)$. A ranking function ($G$) is the important part of a search engine that determines the order in which documents retrieved for a given user query should be presented. The problem of mastering ranking methods is formalized as follows: $G : Q \times D \rightarrow SERP$. Developing reliable ranking techniques may be not easy because user search goals are not static and depend on the search context, e. g. user background knowledge, age, or location, or their specific search intents [31, 150, 193]. Context-aware search adapts SERP using available contextual information ($C$) as follows: $G : Q \times D \times C \rightarrow SERP$. While personalized search considers individual users long and/or short histories [25, 28, 210], context-aware search focuses on short histories to reason about current user situation. Contextual ranking techniques adapt characteristics about groups of users [247], or behavioral changes throughout the whole search engine audience [145, 148, 185, 186]. Nowadays, searching behavior is no longer limited to $\langle Q, SERP \rangle$ paradigm. For instance, the use of intelligent assistants is bringing new challenges [123, 153, 245]. One way to improve quality of searching behavior is to predict user satisfaction with SERP upfront. User satisfaction is widely adopted as a subjective measure of search experience. Kelly

[131] proposes a definition: *satisfaction can be understood as the fulfillment of a specified desire or goal.* Contextual information can be utilized to make user satisfaction prediction for searching more accurately.

### 1.1.4 Context Integration Strategies

Looking from the perspective of the ranking methods, there are different ways in which context can be used in the different web applications. We use the following five strategies for integration of contextual information initially presented in [231]:

1. *Feature normalization*: context is used to normalize the primary context-sensitive features, prior to using the prediction model. The purpose is to process context-sensitive features in a way that reduces their sensitivity to the context.

2. *Feature weighting*: context is used to weight the primary features, prior to prediction. The goal of weighting is to assign more importance to features that, in a given context, are more useful for prediction.

3. *Feature expansion*: the feature space composed of primary features can be expanded with contextual features. The contextual features can be treated by a learning process in the same manner as the primary features.

4. *Model selection*: the prediction can proceed in two steps: (1) first selecting a specialized predictive model based on the context information, (2) then applying this model to the primary features.

5. *Model adjustment*: the prediction can proceed in two steps: (1) first training using only the primary features, (2) then making an adjustment to the prediction based on the context.

## 1.2 Research Questions and Outline

Above, we introduced our main research problem, which we break down into four general research questions.

### 1.2.1 Useful Contextual Information

We start by defining what is useful contextual information for the prediction process:

> **RQ 1**: *What are the general characteristics of useful contextual information?*

We investigate **RQ 1** in Chapter 2, where we introduce the notions of useful context and optimal contextual models, which we call the contextual principle. Essentially, the definition of useful context captures the usual operational situation in which no global optimum is sought, but there is a current system (captured

by some model) that we seek to improve in terms of prediction quality by taking context into account. We introduce the contextual principle which shows that the problem of finding the best model for every test instance can be solved by considering the sub-problems of finding optimal models for subsets of test instances belonging to the corresponding contexts. This is a technical result of a desirable property that allows us to work on *customization* to user types or profiles, or personas, rather than *personalization* to specific individuals. In practice, finding an optimal model for each context can be as hard as finding an optimal model for the complete dataset. Hence, it is usually the case that the type of the predictive model is chosen in advance based on the overall goal of an application, e. g. Markov models for predicting next user action in web sessions.

### 1.2.2 Explicit Contextual Information

Next, we look at the impact of specific contextual aspects, starting with explicit context provided in the web logs such as geographical location, time, and user agent data:

> **RQ 2**: *How to identify useful contextual information from the available list of explicit contexts?*

While users *browse* in a web application, they do not explicitly express their needs, e. g. through queries. However we can understand their demands reasoning based on their behavioral patterns and provided additional contextual information. Web logs contain various additional information about users such as user geographical location, detailed information about user device and its operation system, timestamps etc. We desire to understand if this information is useful to make contextual predictions. The geographical location of users is one of the prototypical examples of contextual information. In the literature, it was shown that the *user's location* is useful contextual information in many applications [17, 203, 238]. A context based on geographical location can have different levels of granularity like continent, country, city [58, 59, 146] and so on.

We start to investigate **RQ 2** in Chapter 2 where we consider it for browsing behavior. Chapter 2 presents our experiments with StudyPortals[2], where we consider a browsing type of task—user next action prediction [256]. It helps to foresee user preferences for adapting content if needed. We build contextual Markov models which are consistent with our contextual principle. We concentrate on a continent level of geographical location due to limitations from the data size side. We use IP addresses as a source to derive geographical location associated with each user session. We have shown in Chapter 2 that for the case of StudyPortals the geographical location is not a useful context according to contextual principle. The possible explanation is that the general audience of StudyPortals consists of students with a certain level of education, approximately same age and having a high level of English. We showed that geographical location might not be universally helpful context.

[2] `http://www.studyportals.eu/`

We continue the investigation of **RQ 2** in Chapter 3 where we consider broader types of explicit contexts. Chapter 3 presents our experiments with a Booking.com service, which allows for finding travel destinations based on users' preferred activities (e. g. 'hiking', 'beach' etc.)—Destination Finder. This service has both browsing and searching aspects. Users search for holiday activities and the service returns a ranked list of recommended destinations. This process is a complex exploratory recommendation task. Such applications are not yet widely available. The Destination Finder can be considered as the application for complex exploratory recommendations, the examples of its use is presented in Figure 1.2. Exploratory recommendation tasks are characterized as follows: users do have preferences what kind of activities they want to do on holidays but they do not have particular information needs as in standard search tasks, when users run the particular query (e. g. 'Find cheapest hotels in Pisa') and expect to find an answer with minimum efforts. Users expect that the system would return a ranked list of reasonable suggestions to explore. For this kind of services, we cannot rely on the user history because it is usually not available (e. g. a user is new for the system or a user changes his preferences). Therefore personalization approaches cannot be used. We called this phenomenon—*Continuous Cold Start Problem* (*CoCoS*). To improve the ranking of retrieved destinations, we propose a method that builds *useful contextual user profiles* that follows our contextual principle. We use explicit contextual data such as user agent data and timestamps to create profiles. Further, we show how contextual profiles help to customize the ranking of search destinations by running online experiment with the real user traffic of Destination Finder.

We present methods how to select useful contexts from the general explicit contextual information that is available in web logs e. g. user location, user agent data, and timestamps.

### 1.2.3 Implicit Contextual Information

Next, we look at the impact of behavioral trails of searching and browsing actions as implicit contextual aspects:

> **RQ 3**: *How to discover users' behavioral aspects as contextual information?*

We investigate **RQ 3** for *browsing* behavior also in Chapter 2. User historical behavior, collected by StudyPortals, is given as a log of web sessions corresponding to browsing activities of users. In our case the users' actions are categorized by the type: *searches*, *clicks on ads*, *homepage visits* etc. Users' activities and their possible orderings within user web sessions are summarized as a *user navigation graph*[3]. We desire to understand if there are any groups of nodes in the navigation graph that reflect different types of user behavior. Then we use this knowledge to characterize the users' behavior in order to improve effectiveness of next users' action prediction. In order to achieve our goal we propose to use several machine

[3]The example of navigation graph is given in Figure 2.3


Figure 1.2: Illustrations of how Destination Finder can be used to explore holiday destinations given preferred activities (the screen shots taken in March 2015). The examples of activities in presented in Figure 3.4 and more examples of how Destination Finder use is presented in Figure 3.3. To access the current version the following link can be used `http://www.booking.com/destinationfinder.en-gb.html`.

learning techniques: First, we discover two types of user behavior on a site by grouping the user navigation graph:

- an expert user, who is experienced with the website interface or searches extensively to find required information;
- a novice user, who needs more time to learn about a website or is not interested much in its content.

Second, we discover changes in user intents during one web session while he is browsing a website. In order to achieve this, we develop a method to segment user sessions that is directly maximizing the accuracy of next action prediction. The proposed methods dramatically improve the prediction accuracy of user trails.

As we illustrated before in Figure 1.1 the same search page has different views for mobiles and for desktops. That affects how users behave while searching, e. g. on mobile devices the main ways of interactions are touch gestures and voice control in contrast to mouse movements on desktops [84, 85, 88, 158, 171, 191]. Recently, a new generation of intelligent assistants, powered by voice, such as Apple's Siri, Microsoft's Cortana, Google Now, etc. have become a common feature on mobile devices. A recent study [81], executed by Northstar Research and commissioned by Google, found out that 55% of the U.S. teens use voice search every day and that 89% of teens and 85% of adults agree that voice search is going to be 'very common' in the near future.

We continue our investigation of **RQ 3** for different scenarios of *searching* behavior in Chapters 4, 5, and 6 where we experiment with the voice-controlled intelligent personal assistant–Microsoft Cortana[4]. The main component of these chapters is that we desire to understand which behavioral signals can be used to measure user satisfaction with different scenarios of intelligent assistants use. User satisfaction is widely adopted as a subjective measure of search experience.

More specifically, Chapter 4 presents our findings about types of intelligent assistants use:

- controlling the device, e. g. call a contact, check the calendar, access an application, etc.;
- searching mobile web;
- performing a complex task through a dialogue interaction with the intelligent assistant.[5]

To investigate the difference in use of scenarios we conducted a user study. We concluded that effort is a key component of user satisfaction across the different intelligent assistants scenarios.

We continue to investigate **RQ 3** in Chapter 5 where we dive deeply into understanding of search dialogues. We study if contexts such as touch signals

[4] `http://www.windowsphone.com/en-us/how-to/wp8/cortana/meet-cortana`

[5] Examples of search dialogues are presented in Figure 4.1 on page 78 and Figure 5.1 on page 102.

are important to predict user satisfaction with search dialogues. For our experimentation we use data collected from the user study in Chapter 4. To conclude if these contexts are useful we use our general analytical framework from Chapter 1.

Finally, we investigate **RQ 3** in Chapter 6 for the mobile web search scenario. There are many cases where a user may not click on any results on SERP but still is satisfied. This scenario is referred to as *good abandonment* [54, 65, 166, 219] and presents a challenge for most approaches measuring search satisfaction. In our experiments we use two data-sources: the user study data from Chapter 4 and real snapshots of web traffic. Similarly, we investigate whether touch signals are useful contexts to detect good abandonment on mobile devices.

We present methods to discover and to evaluate the impact of implicit contextual information such as type of user of user behavior, hidden components of user satisfaction, and touch gestures that are specific for different types of search tasks on mobile devices.

### 1.2.4 Dynamic Contextual Information

Finally, we look at behavioral dynamics—changes in aggregated user behavioral features over time—such as the frequency of query revisions and Satisfied (SAT) and Dissatisfied (DSAT) clicks to detect changes in user satisfaction and drifts in query intent:

> **RQ 4**: *How to define and to detect changes in user satisfaction with retrieved search results?*

We look at indicators of a drop in user satisfaction due to SERPs trained on historical data becoming outdated with a drift in query intent happening because some implicit context (e. g. news event) or over time. When users struggle to find an answer for query $Q$ they run a follow-up query $Q'$ that is an expansion of $Q$. *Query reformulation* is the act of submitting a next query $Q'$ to modify a previous $SERP$ for a query $Q$ in the hope of retrieving better results [98]. Such a query reformulation is a strong indication of user dissatisfaction [9]. We call this the *reformulation signal*. Our hypothesis is that a decrease in user satisfaction with $\langle Q, SERP \rangle$ correlates nicely with the reformulation signal. In other words, the probability of reformulating $Q$ will grow dramatically.

We start to investigate **RQ 4** in Chapter 7. We propose an unsupervised approach, called *Drift Detection in user SATisfaction (DDSAT)*, for detecting drifts in user satisfaction for pairs $\langle Q, SERP \rangle$ by applying the concept drift technique [80, 202, 241, 252] leveraging the reformulation signal. Concept drift primarily refers to an online supervised learning scenario when the relation between the input data and the target variable changes over time [80]. Furthermore, the reformulation signal is considered to be less noisy and if reformulations are fresh and done only by users' initiative then we can say that a reformulation signal is not biased by information coming from the search engine. We conduct a large-scale evaluation using search log data from Microsoft Bing[6]. Our experiments show that the algorithm *DDSAT* works with a high accuracy.

[6] `http://bing.com/`

We continue our investigation of **RQ 4** in Chapter 8, where we extend our method by taking into account more signs of user frustration (lack of search satisfaction) such as: a rate of search abandonment, a dramatic change in query volume, a lowering in average click positions. We conducted a large-scale evaluation with one year of search log data from Yandex[7]. Moreover, our framework outputs the list of drift terms and the list of $URLs$, which can be used for the future re-ranking of $SERP$. The algorithm of the drift detection in user satisfaction can be incorporated in many search-related applications where freshness is required, e. g. in recency ranking, query auto-completion.

In this section we presented the main research questions that we investigated in this dissertation. Next, we will list of the main contributions.

## 1.3 Main Contributions

In this section we summarize the main scientific and practical contributions of this dissertation.

### 1.3.1 Scientific Contributions

The scientific contributions include the following:

- A contextual principle that can be used to preselect useful contextual information offline (Chapter 2).
- An approach to predict user intent switch during a web session (Chapter 2).
- An approach to identify the user type in terms of their expertise level based on interaction behavior (Chapter 2).
- A first characterization of the continuous cold start recommendation problem (*CoCoS*) (Chapter 3).
- An unsupervised approach for discovering and using contextual user profiles for complex exploratory recommendations (Chapter 3).
- The design of the first ever user study to collect interaction signals for search dialogues for voice-controlled intelligent assistants using realistic tasks from web logs (Chapter 4).
- A definition of user satisfaction with search dialogues on intelligent assistants (Chapter 5).
- A an effective method to predict user satisfaction with search dialogues on intelligent assistants using user interaction signals (Chapter 5).

[7] `http://yandex.ru/`

- A an improved method to detect "good abandonment" (queries with satisfying responses yet no further interaction) on mobile devices using user interaction signals (Chapter 6).
- A first unsupervised methods to detect changes in user satisfaction on the SERP-level based on behavior dynamics (Chapter 7 and 8 ).

### 1.3.2 Practical Contributions

All research in this dissertation was conducted using real data from a range of different e-commerce and search applications, in direct collaboration with industry, having direct and indirect impact on their products and services. There practical contributions include:

- A ranker for contextual travel recommendations for Destination Finder that gave significant gains compared to the production baseline at that time: 20% on click-through, and 21% on clicks per user (Chapter 3).
- An improved user satisfaction metric that was adopted for evaluation of Microsoft Cortana search dialogues (Chapter 5).
- A metric that was adopted in production settings of Microsoft Cortana to evaluate good abandonment for mobile web search (Chapter 6).
- A monitoring mechanism to detect outdated SERPs that was adopted as feature by Microsoft Bing (Chapter 7) and Yandex (Chapter 8).

We presented the the list of main contributions of this dissertation. Next, we will outline our research methodology.

## 1.4 Research Methodology

Our general research methodology is to conduct research on realistic data from various online search services, ensuring our results transfer to operational cases. This has a number of important consequences:

**Experimental Datasets** First, wherever possible we use real-world data from Internet companies in our experiments, including: StudyPortals, Microsoft Bing, Booking.com, Yandex, and Microsoft Cortana. In addition, we also use the data from TREC Contextual Suggestion Track 2014[8] [59, 146]. The data collection TREC Contextual Suggestion Track 2015[9] is motivated by our study with Booking.com [61].

**Modeling Framework** Second, in terms of the modeling framework we develop theoretical approaches that can be implemented in a production environment. Therefore, the suggested methods are highly efficient. We embed the developed

[8] https://sites.google.com/site/treccontext/trec-2014
[9] https://sites.google.com/site/treccontext/trec-2015

techniques into prototypes to test them through an evaluation circle (discussed next). In order to facilitate the subsequent refinement of the proposed methods, we build prototypes that can be improved in an iterative manner.

**Evaluation Circle** Third, evaluation is key in operational environments, and we follow the current state of the art approach to evaluation, consisting of:

- *Offline stage*—where progressive evaluation (time-wise) or cross validation (object-wise) is used. For our study with StudyPortals, we use historical data for performing cross validation to derive final evaluation metrics;
- *Online stage*—where the developed techniques are integrated into web systems to test them on real user traffic. For our study of explorative recommendations with the Destination Finder, we employ A/B and multivariate testing procedures providing reliable estimates of the performance of the alternative approaches [155, 223];
- *User studies*—where the data is gathered during the lab study. For our intelligent assistants studies, we construct the simulated tasks so that user study participants could relate to them and they would provide *enough imaginative context* [38]. The data collected in the user study is of high quality since users could directly provide information about the tasks.
- *Crowdsourcing*—where data is collected by employing human judgments procedures, which is a common approach to collecting labeled data for search tasks [240]. For our studies with search engines (Microsoft Bing, Yandex and mobile Microsoft Cortana), we employ judges to label snapshots of data from the real traffic. The labeled data is used to calculate final evaluation metrics.

This section described our general research methodology used in this dissertation. Next, we will present the dissertation overview and original publications on which the dissertation is based.

## 1.5 Dissertation Overview and Origins

This PhD dissertation contributes to the research in predictive web analytics through a series of empirical studies with different online web-based services listed previously in Section 1.4. The studies that comprise the different chapters of this dissertation have been published in peer-reviewed conferences and workshops.

We decided to make each chapter self-contained so it can be read independently of the rest of the dissertation. The consequence is that some chapters have similar discussions of related work.

An overview of the studies included in this dissertation also appeared at the following Doctoral Symposiums:

**[139]** J. Kiseleva. Context mining and integration into predictive web analytics. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 383–388, 2013.

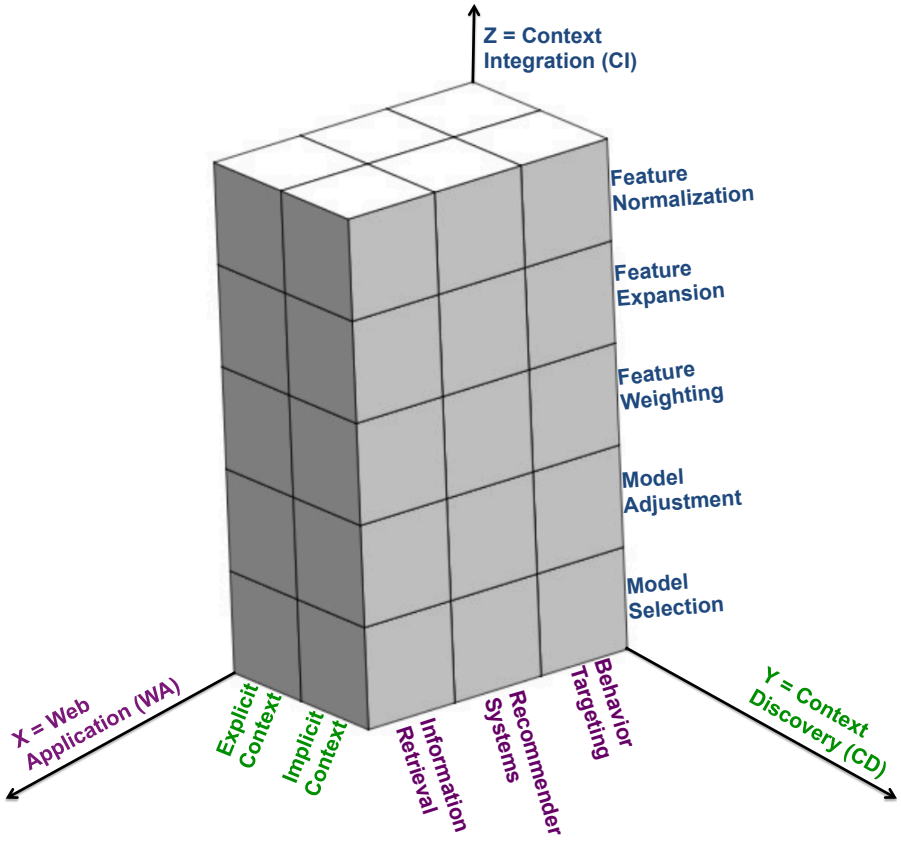



Figure 1.3: A visualization of main components for modeling context-aware systems.

**[140]** J. Kiseleva. Using contextual information to understand searching and browsing behavior. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval (Doctoral Consorcium)*, page 1059, 2015.

The main body of this dissertation is structured in three parts. The first part, comprising Chapters 2 and 3, explores methods to predict user engagement. The second part, comprising Chapters 4, 5, and 6, explores techniques to predict user satisfaction with different intelligent assistants scenarios. The third part, comprising Chapters 7 and 8, explores approaches to predict user satisfaction on SERP-level.

We visualize the main components for modeling context-aware systems in Figure 1.3 that is utilized further to give a structured overview of this dissertation. We distinguish three main components that are represented as a coordinate sys-

tem in Figure 1.3:

- **X = Web Application (WA):** first, it is important to understand the main objectives of web application to make further choices coherently;
- **Y = Context Discovery (CD):** second, modeling choice is to decide how contextual information is defined/discovered;
- **Z = Context Integration (CI):** third, modeling choice is to understand what is the beneficial way to integrate discovered contextual information into the application;

In this dissertation, we study context-aware systems within the scope of the cube presented in Figure 1.3. A context-aware system is not necessarily a point in the presented space. For instance, it is possible that two or more methods are employed for context integration, or the system can be mapped to two or more applications. For example, the Destination Finder serves explorative recommendations so the service can be considered an intersection of recommender systems and information retrieval. Therefore, in general case, a context-aware system is a cube in the defined coordinate system **(WA, CD, CI)**. We would use this abstraction further to project the modeled context-aware system in every chapter to the general picture thus making our contributions clear.

The following publications form the basis of chapters in this dissertation.

- Chapter 2 'User Trails' is based on:

  **[143]** J. Kiseleva, H. T. Lam, M. Pechenizkiy, and T. Calders. Predicting current user intent with contextual markov models. In *Proceedings of the IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 391–398, 2013.

  **[144]** J. Kiseleva, H. T. Lam, M. Pechenizkiy, and T. Calders. Discovering temporal hidden contexts in web sessions for user trail prediction. In *Companion Proceedings of the International Conference on World Wide Web (TempWeb)*, pages 1067–1074, 2013.

  The modeled context-aware system is:

  **(WA)** *behavioral targeting*: the goal is to predict next user action on a website to foresee how a user can be engaged;

  **(CD)** – *explicit context*: the user geographical location;
  – *implicit context*: the technique to discovering the type of user behavior: a novice or an expert user for this website;

  **(CI)** *model selection*: the set of local contextual models are built and a suitable contextual model is selected for online prediction.

- Chapter 3 'Contextual User Profiles' is based on:

**[29]** L. Bernardi, J. Kamps, J. Kiseleva, and M. J. I. Müller. The continuous cold start problem in e-commerce recommender systems. In *Proceedings of the Workshop on New Trends on Content-Based Recommender Systems co-located with ACM Conference on Recommender Systems*, pages 30–33, 2015.

**[149]** J. Kiseleva, M. J. I. Müller, L. Bernardi, C. Davis, I. Kovacek, M. Stafseng Einarsen, J. Kamps, A. Tuzhilin, and D. Hiemstra. Where to go on your next trip? optimizing travel destinations based on user preferences. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 1097–1100, 2015.

**[151]** J. Kiseleva, A. Tuzhilin, J. Kamps, M. J. I. Müller, L. Bernardi, C. Davis, I. Kovacek, M. Stafseng Einarsen, D. Hiemstra, and M. Pechenizkiy. Ranking travel destinations with contextual user profiles. Under Submission, 2016.

The modeled context-aware system is:

**(WA)** *intersection of recommender system and information retrieval*: the goal is to retrieve a ranked list of holiday destinations given the activities pre-selected by a user;

**(CD)** *implicit context*: the available explicit contextual information (user agent data and timestamps) is used to build useful contextual profiles;

**(CI)** *model selection*: an incoming user is mapped to on the contextual profile and it activates a particular contextual ranker.

- Chapter 4 'Intelligent Assistants' is based on:

**[153]** J. Kiseleva, K. Williams, J. Jiang, A. H. Awadallah, I. Zitouni, A. Crook, and T. Anastasakos. Understanding user satisfaction with intelligent assistants. In *Proceedings of the ACM SIGIR Conference on Human Information Interaction and Retrieval (CHIIR)*, pages 121 – 130, 2016.

The modeled context-aware system is:

**(WA)** *information retrieval*: the goal is to understand main components user satisfaction for the three scenarios of user interaction with intelligent assistant: controlling a device, mobile web search, and search dialogues;

**(CD)** *implicit context*: the list of implicit contexts are gathered by conducting the user study:

  - *general across scenarios:* effort users spent, speech recognition quality;
  - *specific to the web search:* number of queries leading to satisfaction, source of user satisfaction (e. g. the knowledge graph answer, the image answer, SERP, or visited website),

  - *specific to search dialogues:* graded user satisfaction with subtasks.

**(CI)** *model adjustment*: we look how understanding of user satisfaction can be adjusted using information about the scenario type.

- Chapter 5 'Search Dialogues' is based on:

  **[152]** J. Kiseleva, K. Williams, A. H. Awadallah, I. Zitouni, A. Crook, and T. Anastasakos. Predicting user satisfaction with intelligent assistants. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, 2016.

  The modeled context-aware system is:

  **(WA)** *information retrieval*: the goal is to predict user satisfaction with search dialogues on intelligent assistants.

  **(CD)** *implicit context*: the touch-based and voice-based features are inferred from the web logs;

  **(CI)** *feature expansion*: the feature set is expanded with contextual information to train a predictor.

- Chapter 6 'Good Abandonment' is based on:

  **[245]** K. Williams, J. Kiseleva, A. C. Crook, I. Zitouni, A. H. Awadallah, and M. Khabsa. Detecting good abandonment in mobile search. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 495 – 505, 2016.

  **[244]** K. Williams, J. Kiseleva, A. Crook, I. Zitouni, A. H. Awadallah, and M. Khabsa. Is this your final answer? evaluating the effect of answers on good abandonment in mobile search. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, 2016.

  The modeled context-aware system is:

  **(WA)** *information retrieval*: the goal is to detect good abandonment during mobile web search;

  **(CD)** *implicit context*: the touch-based and answer types (e. g. image, knowledge graph etc. ) features are inferred from the web logs;

  **(CI)** *feature expansion*: the feature set is expanded with contextual information to train a detector.

- Chapter 7 'Query Reformulations' is based on:

  **[145]** J. Kiseleva, E. Crestan, R. Brigo, and R. Dittel. Modelling and detecting changes in user satisfaction. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*, pages 1449–1458, 2014.

The modeled context-aware system is:

**(WA)** *information retrieval:* the goal is to detect changes in user satisfaction with $\langle Q, SERP \rangle$ over time;

**(CD)** *implicit context*: the change in user intent is happening due to implicit context (e. g. news event). It is discovered using a proxy such as the reformulation signal which is probability to reformulate a query measured over some period of time;

**(CI)** *model adjustment*: the detected change is the signal to adjust the SERP ranking.

- Chapter 8 'Failed SERPs' is based on:

  **[148]** J. Kiseleva, J. Kamps, V. Nikulin, and N. Makarov. Behavioral dynamics from the SERP's perspective: What are failed SERPs and how to fix them? In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*, pages 1561–1570, 2015.

  The modeled context-aware system is:

  **(WA)** *information retrieval*: the goal is to detect if $\langle Q, SERP \rangle$ starts to fail in some moment of time;

  **(CD)** *implicit context*: the reformulation signal, the rate of search abandonment, the dramatic change in query volume, the lowering in average click positions that are inferred from web logs;

  **(CI)** *model adjustment*: the detected change is the signal to adjust the SERP ranking.

- **Chapter 9** concludes the dissertation. We revisit the research questions and gives directions for future research.

The research for the publications that constitute Chapters 2, 3, 4, 5, 7, and 8 was conducted by Julia Kiseleva as a first author. First authorship is shared with Kyle Williams on the conference publications [244, 245] that make up Chapter 6.

**Other Publications** In addition to the publications included in this dissertation, we also published several other papers over the course of this PhD project, as listed below.

**[60]** A. Dean-Hall, C. L. A. Clarke, J. Kamps, and J. Kiseleva. Online evaluation of point-of-interest recommendation systems. In *Proceedings of SCST@ECIR*, 2015.

**[61]** A. Dean-Hall, C. L. A. Clarke, J. Kamps, J. Kiseleva, and E. M. Voorhees. Overview of the TREC 2015 contextual suggestion track. In *Proceedings of the Text REtrieval Conference (TREC)*, 2015.

**[93]** H. Hashemi, C. L. A. Clarke, A. Dean-Hall, J. Kamps, and J. Kiseleva. On the reusability of open test collections. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 827–830, 2015.

**[94]** S. H. Hashemi, C. L. A. Clarke, A. Dean-Hall, J. Kamps, and J. Kiseleva. An easter egg hunting approach to test collection building in dynamic domains. 2016. EVIA@NTCIR.

**[146]** J. Kiseleva, A. Montes García, Y. Luo, J. Kamps, M. Pechenizkiy, and P. De Bra. Applying learning to rank techniques to contextual suggestions. In *Proceedings of the Text REtrieval Conference (TREC)*, 2014.

**[147]** J. Kiseleva, J. Kamps, and C. L. A. Clarke. Contextual search and exploration. *Communications in Computer and Information Science*, 2015.

**[150]** J. Kiseleva, A. Montes García, J. Kamps, and N. Spirin. The impact of technical domain expertise on search behavior and task outcome. In *Proceedings of WSDM Workshop on Query Understanding and Reformulation for Mobile and Web Search (QRUMS)*, 2016.

**[162]** H. T. Lam, J. Kiseleva, M. Pechenizkiy, and T. Calders. Decomposing a sequence into independent subsequences using compression algorithms. In *Proceeding of the ACM SIGKDD Workshop on Interactive Data Exploration and Analytic (IDEA)*, pages 67–75, 2014.

**[218]** N. Spirin, M. Kuznetsov, J. Kiseleva, Y. Spirin, and P. Izhutov. Relevance-aware filtering of tuples sorted by an attribute value via direct optimization of metrics. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 979–982, 2015.

# Part I

# Predicting User Engagement

# Part I
# Predicting User Engagement

In the first part of this dissertation, we explore methods to predict user engagement. We present methods to select useful contexts from the general explicit contextual information that is available in web logs, e. g. user location, user agent data, timestamps, and implicit contextual information such as click and browsing behavior. Specifically, Part I deals with the following research questions:

> **RQ 1**: *What are the general characteristics of useful contextual information?*
> **RQ 2**: *How to identify useful contextual information from the available list of explicit contexts?*
> **RQ 3**: *How to discover users' behavioral aspects as contextual information?*

Chapter 2 investigates **RQ 1** by introducing the notions of useful context and optimal contextual models, which we call the contextual principle; **RQ 2** by experiments on browsing behavior at StudyPortals focusing on the impact of geographic context; and **RQ 3** focusing on navigation actions in session: searches, clicks on ads, homepage visits etc. Chapter 2 is based on [143, 144].

Chapter 3 continues the investigation of **RQ 2** by considering broader types of explicit contexts, such as time and user agent data, in experiments at Booking.com and their Destination Finder service. Chapter 3 is based on [29, 149, 151].

# 2 User Trails

In many web sites like e-shops and information portals, predictive modeling is used to understand user intentions based on their browsing behavior. User behavior is inherently sensitive to context. Identifying the relevant context can help to improve the prediction performance. In this chapter, we propose a formal approach in which the context discovery process is defined as an optimization problem. For simplicity, we assume a concrete yet generic scenario in which context is considered to be a secondary label of an instance, which is either known from the available contextual attribute (e. g. user location), or can be induced from the training data (e. g. novice vs. expert user). In an ideal case, the objective function of the optimization problem has an analytical form, enabling us to design a context discovery algorithm solving the optimization problem directly. An example with Markov models, a typical approach for modeling user browsing behavior, shows the derived analytical form of the optimization problem. Experiments with a real-world use-case show that we can discover useful contexts allowing us to significantly improve the prediction of user intentions with contextual Markov models.

## 2.1 Introduction

When exploring available or discovered contextual information for predictive modeling, we aim to focus on a subset of *the most promising contexts*. For instance, we can perform data exploration and use domain expertise for choosing an appropriate subset of contexts. However, for complicated and large-scale datasets, deep understanding of data by exploration can be rather limited. Therefore, an alternative straightforward context selection approach is to direct an effectiveness evaluation of every subset of the set of targeted contexts. This solution is computationally demanding when the set of targeted contexts has a high cardinality. Moreover, in several cases, evaluation must be done in an online setting with a real information system in operation that is also very expensive and time demanding.

We propose an automatic technique that preselects *a set of useful (or effective)*

*contexts* in offline settings directly by optimizing the effectiveness of a predictive modeling method. This chapter focuses on concrete predictive modeling application—the next action prediction in user session. Therefore, we concentrate on the following main research question:

> *How to automatically discover a set of useful contexts to predict a user's next action in a web session?*

We break down our general research problem into six specific research questions:

> **RQ 2.1** *What is contextual prediction?*

We formulate the definition of useful context discovery from the web logs as an optimization problem. Even when domain experts are not available and data exploration gives only partial knowledge about the data and if direct context evaluation and testing are expensive, it is still possible to select a good set of contexts if we know the closed form of the objective function of the optimization problem. On the one hand, an analytical form of the objective function provides us with useful mathematical insights of the problem. It may give us a good hint for context discovery even in the case that the optimization problem is hard. On the other hand, it enables us to evaluate the contexts in an offline setting before performing an online testing with a small set of selective contexts. For simplicity, but without loosing the generality of our study we assume that all the contexts are non-overlapping and that a user is at any moment associated only with one contextual category.

> **RQ 2.2** *What machine learning methods can be used to discover useful context for the next action prediction?*

We focus on Markov models that are commonly used for modeling web user behavior. Our analysis shows that the objective function calculated as the expected accuracy of prediction using the Markov model has a closed form. Analyzing the analytical form of the objective function helps us to find interesting properties of adopting contextual information for prediction with the Markov model. Namely, *if the data are generated by a Markov model, any context preserving the Markovian property in each contextual category is useful in the sense that the accuracy of prediction using Markov model built for each category of the context is at least as large as the expected accuracy of the Markov model built for the whole data.* This property is a theoretical justification of context-aware method for prediction using Markov models.

> **RQ 2.3** *What is the impact of geographical location as contextual information?*

The geographical location of users is one of the prototypical examples of contextual information. In the literature, it was shown that the *user's location* is useful contextual information in many applications [17, 203, 238]. A context

based on geographical location can have different levels of granularity like continent, country, city and so on. In this section, we use geographical location as a explicit context and demonstrate how our methodology can be applied to check if it is useful context.

**RQ 2.4** *How to discover types of user behavior based on performed actions?*

The type of user behavior is important information [9]. This information whether a user is very well familiar with a particular website's functionality or falls into the category of novice users is not available explicitly. We propose a context discovery approach that is able to infer information about user expertise automatically based on the type of actions performed by a user.

**RQ 2.5** *How to discover changes of user intent within a web session?*

We introduce two contextual prediction methods to segment the web session as demonstrated Figure 2.1. Let us consider for example how contextual information about user intents can be applied for the next action prediction. Users usually perform actions in a sequential manner on a website and the set of all actions is given. The goal is to predict the next action each user will perform given historical data about user activities. For instance, in Figure 2.1, we present a web session as a sequence of user's actions $(a, b, f, d, c, a, a)$ (from the alphabet $\{a, b, c, d, e, f\}$). We assume that under one context users perform a specific set of actions and when the context is switched to another one, another set of actions is performed. That means a context is defined as an external factor which is associated with a specific set of user actions. For instance, in a web session, there are sets of actions associated with the context like 'search' while there are other sets of actions associated with the context like 'buying'.

**RQ 2.6** *How effective is our approach to discover useful contexts on a realistic sample of traffic?*

We validate our approach on the real sample of user traffic from StudyPortals. Our experiments illustrate that (1) if useful contexts are discovered, the local Markov models predict user intentions statistically significantly better than the global model, and (2) local Markov models stil perform well, i. e. not significantly worse than a global model, even when the contexts are absolutely not useful and have substantially smaller number of instances to induce local models.

The remainder of this chapter is organised as follows. In Section 2.2 we introduced related work on context-awareness in supervised learning applications. In Section 2.4 we introduce definitions of context-aware predictive analytics. In Section 2.5 we introduce a specific case of using contextual information for improving prediction ability of Markov models. In Section 2.6 we propose the context discovery method based on navigation graph clustering. We present our experimental study using a real Web portal data sets in Sections 2.7 and 2.8. Section 2.9 concludes.

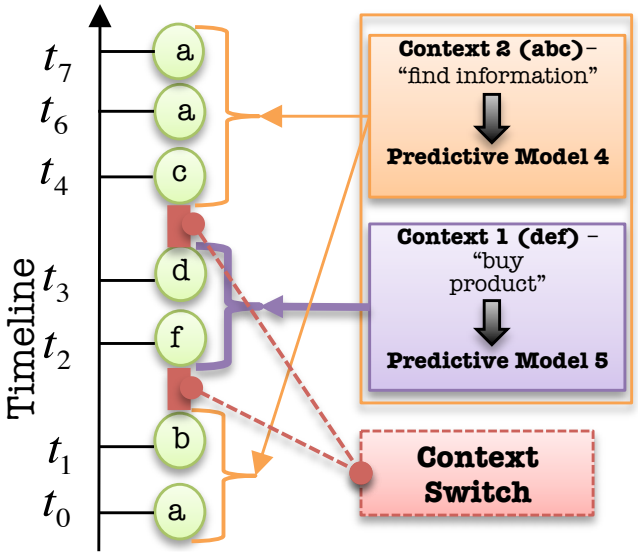



Figure 2.1: An example of the switch in user intent within one session: ($a$, $b$, $f$, $d$, $c$, $a$, $a$).

## 2.2 Background and Related Work

In this section, we will discuss related work relevant to the research described in this chapter, covering two broad strands of research. First, we discuss research on how contextual information is used for different predictive modeling tasks in Section 2.2.1. Second, we present overview of how Markov Models are applied for predictive web analytics in Section 2.2.2.

### 2.2.1 Integrating Context in Predictive Modeling Tasks

Many studies have demonstrated that integrating context-awareness into predictive modeling helps to better understand user information in computational advertising [18, 46], recommender systems [4, 180, 189], web search [43] and other related areas [247]. According to Prahalad [183], a context has temporal (when), spatial (where), and technological (how) dimensions.

In many web applications, contextual information is available along with the data. For instance, the contextual information is provided explicitly in the form of additional features describing, e. g. user current location, user device or gender. These *explicit* contexts are different from *implicit* contexts that can be only discovered from data [233, 254]. In terms of interactive systems, Palmisano et al. [180] has shown that the previous history of user interaction with the system should be considered and especially changes in user behavior. Zliobaite [253] captured context specific information by forcing independence between contexts and class labels. The context was defined as an artifact in the data that does not

directly predict the class label, e. g. accent in speech recognition. Mazhelis et al. [175] considered approaches of using explicit and implicit contextual information for local model selection. Later, Zliobaite et al. [254, 255] proposed context-aware systems as two level prediction models for food sales. Each product is classified it to one of the predefined categories based on structural properties of the sales time series, then a category specific predictor is applied to predict the future sales.

In machine learning, context was considered as *contextual features* in supervised concept learning [232]. Contextual features do not determine or influence the class of an object directly, but improve predictive performance when used together with other predictive features [233]. Recent approaches consider how to derive such features. The contextual features are useful for classification only when they are considered in combination with other features. For example, in medical diagnosis problems, the patient's gender, age, and weight are often available. These features are contextual, since they (typically) do not influence the diagnosis when they are considered in isolation. The context can be used to split users into subgroups sharing similar backgrounds. Users in the same interest group or with the same intent usually behave in a similar way. Therefore, user intentions are easier to recognize when the predictive models cleverly leverage the available contextual information, e. g. employing local models for each context.

### 2.2.2 Markov Models for Predictive Web Analytics

Several Markov models were proposed for modeling user web data: first-order Markov models, hybrid-order tree-like Markov models [68], prediction by partial match forest [47], $k$th-order Markov models [63], variable order Markov models (VOMM) [37] that provide the mean to capture both large and small order Markov dependencies. It was shown in [50] on a large data set that it is better to use the variable order Markov models for this purpose. Other, perhaps the most commonly used techniques, are based on Hidden Markov Models (HMM). However, working with HMMs typically requires understanding of the domain and very large training samples [26].

Very recently, efforts have been made in modeling session search using the Partially Observable Markov Decision Process (POMDP) [172, 173, 248]. This line of research investigated the best ways to design the states, actions, and rewards within a POMDP framework for complex information retrieval tasks.

In this chapter, we defined a contextual principle that is a technique to discover useful contexts, that is formulated as optimization problem. As predictive models we used Markov models.

To summarize, the key distinctions of our work compared to previous efforts are: we introduced a definition of useful contexts for predictive web analytics tasks; we enhance the existing user modeling methods based on Markov models by introducing the contextual Markov models, and our context discovery technique (contextual principle) is novel as it formulate as an optimization problem.

## 2.3 Preliminaries and Notations

This section will introduce the preliminaries and notations:

**Event** $(a, t)$**.** Given a set $A = \{a_1, \ldots, a_m\}$ of event types (all possible actions users can perform), let an *event* be a pair $(a, t)$, where $a \in A$ is an event type and $t$ is the occurrence time of event.

**Web Session** $s$**.** Let $s$ be a web session of the user is an ordered sequence of events:

$$s = \langle (a_1, t_1), (a_2, t_2), \ldots, (a_n, t_n) \rangle \tag{2.1}$$

such that $a_i \in A$ for all $i \in [1, n]$ and $t_i \in [s_s, s_e]$, $t_i < t_{i+1}$ for all $i \in [1, n-1]$, where $s_s, s_e$ are integers denoting the starting and ending time of the session. Note that we do not have $t_i = t_{i+1}$, i. e. several events cannot occur at the same time.

**Web Log** $D$**.** Let $D = \{s_1, s_2, \ldots, s_n\}$ be the general representation of users' historical behavior is given as a log with web sessions.

**Contextual Space** $\Theta$**.** Let $\Theta = C_1 \times C_2 \times C_3 \times \cdots \times C_N$ be the space of all possible contextual features associated with every data instance, where each $C_i$ is a context.

**Contextual Feature** $C$**.** Let $C$ be a contextual feature with $n$ categories: $C = \{c_1, c_2, \ldots, c_n\}$ associated with each data instance $s \in D$. For example, *time of the day* is the contextual feature with the four categories $\{c_1 = \text{morning}, c_2 = \text{afternoon}, c_3 = \text{evening}, c_4 = \text{night}\}$.

**Contextual feature vector** $\theta_s$**.** Let us denote $\theta_s \in \Theta$ as the contextual feature vector associated with sequence $s$.

**Decision space** $V$**.** Let $V$ be a decision space for our predictive model. For example, in the case which predicts the next activity which the user will perform, the decision space $V$ is the same as the the set of events type $A$, i. e. $V \equiv A$.

**Predictive model** $M$**.** Let $M : \Theta \times D \mapsto V$ be a predictive model that maps each test sequence $s \in D$ associated with the contextual information $\theta_s$ to the decision space $V$.

**Evaluation Function** $F$**.** Let $F(s, M(\theta_s, s)) : D \times V \mapsto R$ be the function evaluates how good a model is. An example of the evaluation function is the number of true predictions made by $M$ over the test instance $s$. For instance, assume that the model $M$ predicts $s = ababc$ as $M(\theta_s, s) = \underline{ab}ed\underline{c}$ then it makes three true predictions corresponding to the underlined activities, i. e. $F(s, M(\theta_s, s)) = 3$.

**Test Set** $T$**.** Let $T \subseteq D$ be a set of test instances and denote $Pr(s)$ as the probability that $s \in T$.

**Evaluation Metric** $E$**.** To evaluate the performance of the predictive model $M$ over the test set $T$ we calculate $E[T, M] = \sum_{s \in T} Pr(s) \cdot F(s, M(\theta_s, s))$.

## 2.4 Contextual Prediction

This section discusses generalized definitions of contextual predictive analytics to given an answer to our **RQ 2.1** *What is contextual prediction?*

As a running example, we consider a website containing five different activities with categorical labels $a, b, c, d$ and $e$. Every user visiting the website produces a sequence of transition activities corresponding to the categories that the user has visited. In this example, $D$ is the set of all possible sequences of activities from the categories $a, b, c, d$ and $e$. The example of the possible web session is presented in Figure 2.1. As an additional information we consider an available contextual features.

To simplify the discussion, we consider contextual feature that have only two categories. The discussion of the general cases with more than two categories is very similar. Assume that we have a context $C$ with two categories $c_1$ and $c_2$ dividing the test set into two disjoint subsets $T_1$ and $T_2$ such that $T = T_1 \cup T_2$. Denote $M_1$ and $M_2$ as two predictive models built for the categories $c_1$ and $c_2$ respectively. Let $P(c_1)$ and $P(c_2)$ be probabilities that a test instance belonging to the category $c_1$ and $c_2$ respectively. The value of the expectation $E[T, M]$ can be considered as an objective that we need to optimize and assume that $M^*$ is the optimal model, i. e. $M^* \arg\max_M E[T, M]$. We formulate the following proposing called Contextual Principle under the described assumptions .

**Proposition 1** (Contextual Principle)**.** *Let $M^*$ be an optimal model on $T$ then it is a combination of $M_1^*$ and $M_2^*$. Where $M_1^*$ is an optimal model for $T_1$ and $M_2^*$ is an optimal model for $T_2$.*

*Proof.* Because $M_1^* = \arg\max_{M_1} E[T_1, M_1]$ and $M_2^* = \arg\max_{M_2} E[T_2, M_2]$ we must have $E[T_1, M_1^*] \geq E[T_1, M^*]$ and $E[T_2, M_2^*] \geq E[T_2, M^*]$. We further derive:

$$P(c_1)E[T_1, M_1^*] \geq P(c_1)E[T_1, M^*] \tag{2.2}$$
$$P(c_2)E[T_2, M_2^*] \geq P(c_2)E[T_2, M^*] \tag{2.3}$$

$$P(c_1)E[T_1, M_1^*] + P(c_2)E[T_2, M_2^*] \geq E[T, M^*] \tag{2.4}$$

On the other hand, since $M^* = \arg\max_M E[T, M]$, we have:

$$E[T, M^*] \geq P(c_1)E[T_1, M_1^*] + P(c_2)E[T_2, M_2^*] \tag{2.5}$$

From two inequalities 2.4 and 2.5 we imply that: $E[T, M^*] = P(c_1)E[T_1, M_1^*] + P(c_2)E[T_2, M_2^*]$. In other words, $M^*$ is a combination of $M_1^*$ and $M_2^*$. □

Proposition 1 proposes that the problem of finding the best model for every test instance can be solved by considering the sub-problems of finding optimal models

for test instances in each individual contextual category. This result provides us with theoretical judgment for customization and exploitation of contextual information in predictive analytics.

Nevertheless, in practice finding an optimal model for each contextual category is usually as hard as finding an optimal model for the whole data. Indeed, it is usually the case that the type of model is chosen in advance, e. g. Markov models. Model's parameters are estimated from training data $D$. Under this circumstance, contextual predictive analytics seeks for a context such that it divides the training data into two subsets $D_1$ and $D_2$ and the predictive models trained on $D_1$ and $D_2$ improve the predictive performance in comparison to the model trained on the whole training data. To this end, we define useful contexts as follows:

**Definition 1** (Useful Context)**.** *Given a model $M$ built based upon the whole training data $D$ and $M_1$, $M_2$ are two models built based upon $D_1$ and $D_2$ corresponding to each contextual category of a context $C$ respectively. The context $C$ is useful if and only if: $E[T_1, M_1] \geq E[T_1, M]$ and $E[T_2, M_2] \geq E[T_2, M]$*

**Definition 2** (Contextual Prediction)**.** *The contextual prediction is the special way to design the learning process that restricts the search space of $M$ by defining the following functions:*

- *The mapping between the useful contextual feature and the contextual categories:*
  $\boldsymbol{G} : C \rightarrow \{c_1, c_2, \ldots, c_n\}$*;*
- *The mapping between the contextual categories and the individual learner:*
  $\boldsymbol{H} : c_i \rightarrow M_i$*.*

In this section we presented the answer to our **RQ 2.1** *What is contextual prediction?* by defining explicitly what is contextual prediction. Next, we will describe the machine learning method that is used to predict next user action utilizing contextual information.

## 2.5 Contextual Markov Models

In this section we address our **RQ 2.2** *What machine learning methods can be used to discover useful context for the next action prediction?* by discussing a specific case of using contextual information for improving prediction ability of Markov models. In particular, we are given log of sequences of activities performed by users in a web application. The task is to predict the next activity in a sequence. Markov model is chosen as a predictive model for this problem. We are interested in finding useful context such that Markov models built for each category of the context improve the prediction performance compared to the Markov model built for the whole data. We call this problem the *contextual Markov model*.

To simplify the discussion, we only consider the special case with the first order Markov model or Markov chain. Generalization of our discussion to Markov

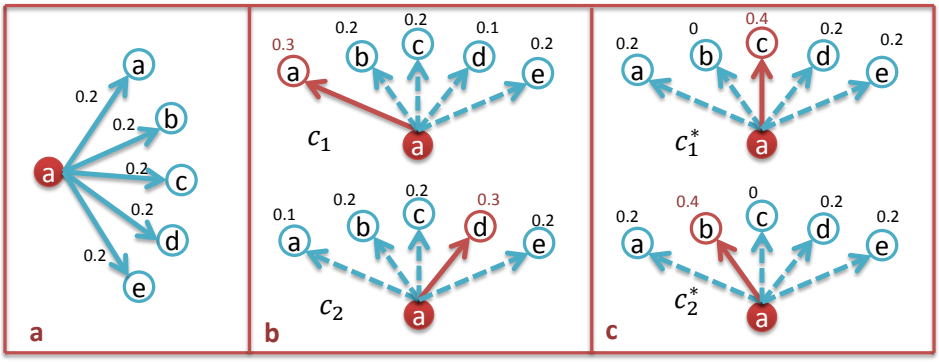



Figure 2.2: An example of transition distributions from $a$ to the other states. Two contexts $C = \{c_1, c_2\}$ and $C^* = \{c_1^*, c_2^*\}$ have different transition distributions. The most probable transition paths are highlighted with red-purple color.

models with any order is similar to that special case. A Markov chain $M$ is associated with a transition probability matrix $[P(a_j|a_i)]$, where $P(a_j|a_i)$ is the probability of transition from the activity $a_i$ to the activity $a_j$.

For any activity $a \in kA$, we denote $m(a)$ as the activity with highest transition probability from the activity $a$, i. e. $m(a) = \arg\max_{b \in kA} Pr(b|a)$. Given that the current state is the activity $a$, if the data follows Markovian property then $m(a)$ is always the best prediction of the next state. Therefore, we consider a predictor which always chooses the most probable transition for the next state. If the test sequences in $T$ are random samples from the Markov model $M$, the expected accuracy of the predictor, i. e. the expectation of true prediction rate can be calculated as follows:

$$E[T, M] \quad = \quad \Sigma_{a \in kA} P(a) P(m(a)|a) \tag{2.6}$$

Let $C = \{c_1, c_2\}$ be any context and $M_1, M_2$ are two Markov chains built for each categories $c_1$ and $c_2$ respectively. Consider a new predictive model that uses $M_1$ to predict test sequences belonging to $T_1$ corresponding to the first category $c_1$ and uses $M_2$ to predict test sequences belonging to $T_2$ corresponding to the second category $c_2$. We also denote $[P_1(a_j|a_i)]$ as the transition matrix of the Markov model $M_1$ and $[P_2(a_j|a_i)]$ as the transition matrix of the Markov model $M_2$. If two test sets $T_1$ and $T_2$ contain randomly sampled sequences from two Markov models $M_1$ and $M_2$ then the expected accuracy of this prediction can be calculated as follows:

$$E[T, M_1, M_2] \quad = \quad P(c_1) E[T_1, M_1] + P(c_2) E[T_2, M_2] \tag{2.7}$$

where $P(c_1)$ and $P(c_2)$ stand for the probability of the test sequence belonging

to the first and the second category respectively and:

$$E[T_1, M_1] = \sum_{a \in kA} P_1(a) P_1(m_1(a)|a) \tag{2.8}$$

$$E[T_2, M_2] = \sum_{a \in kA} P_2(a) P_2(m_2(a)|a) \tag{2.9}$$

**Proposition 2.** *Assume that the test data possess the Markovian property and this property holds for every category of a context $C$. Moreover, the training data $D$ together with $D_1$ and $D_2$ are large enough such that we can learn accurate Markov models $M$, $M_1$ and $M_2$ then that context is useful, i. e.: $E(T_1, M_1) \geq E(T_1, M)$ and $E(T_2, M_2) \geq E(T_2, M)$*

*Proof.* Under the category $c_1$, let $P(m(a), a|c_1)$ be the probability of the event indicating that the current activity is $a$ and the next activity is $m(a)$. We have:

$$E[T_1, M] = \sum_{a \in kA} P(m(a), a|c_1) \tag{2.10}$$

$$= \sum_{a \in kA} P(m(a)|a, c_1).P(a|c_1) \tag{2.11}$$

$$= \sum_{a \in kA} P(m(a)|a, c_1).P_1(a) \tag{2.12}$$

$$\leq \sum_{a \in kA} P_1(m_1(a)|a, c_1).P_1(a) \tag{2.13}$$

$$\leq E[T_1, M_1] \tag{2.14}$$

The inequality $E[T_2, M] \leq E[T_2, M_2]$ can be derived in a similar way from which the proposition is proved. □

Proposition 2 shows that if the data possesses the Markovian property then exploiting any context preserving the Markovian property is always beneficial. This proposition can be considered as a theoretical judgment for using contexts to improve Markov model. In practice, Markov models are usually learnt from training data. The accuracy of model's parameters estimation is highly dependent on the amount of available data. When we exploit a context, the training data is split into smaller portions by the context which may cause the decline in the accuracy of parameter estimation.

Finally, the contextual Markov problem is defined as an optimization problem as follows:

**Definition 3** (Contextual Markov)**.** *Given training data $D$, find the context $C = \{c_1, c_2\}$ splitting $D$ into $D_1$ and $D_2$ such that the Markov models $M_1$ and $M_2$ learnt from $D_1$ and $D_2$ respectively maximize the evaluation function on the test set $T$: $E[T, M_1, M_2]$.*

In this section we answered our **RQ 2.2** *What machine learning methods can be used to discover useful context for the next action prediction?* by introducing contextual Markov models. Next, we will present three methods that use concept of contextual Markov models to predict next user action in the web session.

## 2.6 Techniques for discovering Useful Contexts

In this section we present our approaches to discover useful contexts that are either given (explicit) or needed to be discovered from the data (implicit). In order to illustrate the key idea behind the proposed technique, consider an example in Figure 2.2 where transition probabilities $P(x|a)$ $(x \in \{a, b, c, d, e\})$ from the current state $a$ to the other states are shown. Figures 2.2.b and 2.2.c show the transition probability $P(x|a, C)$ $(x \in \{a, b, c, d, e\})$ in two different contexts $C = \{c_1, c_2\}$ and $C^* = \{c_1^*, c_2^*\}$.

In Figure 2.2.a, all transitions are equally probable. Therefore, the transition probability distribution from $a$ has very high entropy making prediction ineffective. If we use the predictor always predicting the most probable transition, the expected true prediction rate is 0.2.

The situation is changed when we consider two contexts $C$ and $C^*$. In particular, in Figure 2.2.b, the distributions $P(x|a, c_1)$ and $P(x|a, c_2)$ both have lower entropy than the transition distribution $P(x|a)$. Under the context $C$, the true prediction is $P(m(a)|a, c_1) = 0.3$ in the category $c_1$ and $P(m(a)|a, c_2) = 0.3$ in the second category. Similarly, under the context $C^*$ the true prediction rate is $P(m(a)|a, c_1^*) = 0.4$ in the category $c_1^*$ and $P(m(a)|a, c_2^*) = 0.4$ in the second category. Therefore, by exploiting the context $C^*$ we may increase the prediction accuracy from 0.2 to 0.4.

Common sense tells us that the prediction is easier if the context splits the data into homogeneous groups. In doing so, users with similar behavior are grouped together which may result in low-entropy transition distribution. A possible clustering algorithm to group users is an agglomerative hierarchical clustering algorithm which uses the objective function $E[T, M_1, M_2]$ as a principle for merging clusters. According the definition 2, our approach consists of two important components: (1) a *clustering algorithm*, which groups training sequence into groups with similar sequences and (2) an *alignment procedure*, which assigns new test sequence to clusters given partial content of the test sequence being seen so far [215].

### 2.6.1 Using Geographical Location

This section investigates our **RQ 2.3** *What is the impact of geographical location as contextual information?* Our web log contains a user location. In the literature, it was shown that the *users' location* is useful contextual information in many applications [17, 203, 238]. A context based on geographical location can have different levels of granularity like continent, country, city and so on. In our experiments we concentrate on a continent level due to limitations from the evaluation side.

**Grouping session** We use user IP addresses as contextual features, then $\theta_s = IP$ is contextual vector associated with session $s$. We define six contextual categories: $C_{geo} = \{C_1 = Europe, C_2 = Africa, C_3 = North\ America, C_4 = South\ America, C_5 = Asia, C_6 = Oceania\}$.

**Geographical alignment function** $kD$ is divided into six disjoint training sets

associated with the continent: $kD_{Europe}$, $kD_{Africa}$, $kD_{Asia}$, $kD_{NorthAmerica}$, $kD_{SouthAmerica}$, $kD_{Oceania}$.

### 2.6.2 Discovering User Expertise

In this section, we address our **RQ 2.4** *How to discover types of user behavior based on performed actions?* Let us recall that a general representation of users' historical behavior is given as a log of web sessions $kD = \{S_1, S_2, \ldots, S_n\}$ where each web session is a sequence of states $S_i = (a_1, a_2, \cdots, a_m)$ corresponding to historical browsing activities of a user. In our case the users' actions are categorized by the type of the users' actions: *searches*, *clicks on ads* or *homepage visits*. A complete set of used categories is presented in Figure 2.3 as graph nodes. However the set of all possible activity states depends on the needs of a particular service e. g. a visit of the home page can be considered as an activity. Thus, activities and their possible orderings within user web sessions can be summarized as a *user navigation graph*.

**Definition 4** (User navigation graph)**.** *A user navigation graph is a directed and weighted graph $G = (V, E)$, where $V$ is a set of vertices corresponding to all possible user actions $kA$ and $E$ are the set of edges $(a_i, a_j)$. Each edge $e$ of $G$ is associated with a weight $w(e)$ indicating the transition probability between two incident vertices of the edges.*

Depending on user experience they may perform different activities by visiting different states in the navigation graph. Therefore, we propose a user action clustering method based on community detection in the navigation graph. We want to understand if there are any groups of nodes in the navigation graph and then use this knowledge to characterize the users' behavior. Intuitively there are two types of user behavior on a site: (1) "expert" users, who are experienced with website interface or searches extensively to find required information, and (2) "novice" users, who need more time to learn about a website or are not interested much in content

Assume that we have $n$ graph partitions by using a communities detection method. Discovered groups of states may be interpreted after analysis e. g. $V_i$ corresponds to a "novice" user's behavior. However, if $n$ is too big to analyze then we have clusters: $\{V_i\}_{i=1}^{n}$. To simplify the discussion, we consider two clusters: Let $V_{exp}$ and $V_{nov}$. $V_{exp}$ corresponds to states which are visited by an "expert" user (red states in Figure 2.3) and $V_{nov}$ consists of activates related to a "novice" user (green sector in Figure 2.3). Having clusters of states, we can align each web session with corresponding clusters. If a web session contains both red and green states, the alignment is performed by examining the sequence from the left to the right: if an expert state encounters in a sequence then the method assigns the sequence to $V_{exp}$, otherwise, a session is assigned to $V_{nov}$ so far. Let us call this procedure *sequential user alignment*. For example, let $S_i = ceabbbaa$, $V_{exp} = \{a, b\}$ and $V_{nov} = \{c, d, e\}$. The alignment function $A$ aligns the sequence $S_i$ to clusters as follows. First, $A$ sees the state $c$ and $S_i$ is temporarily assigned

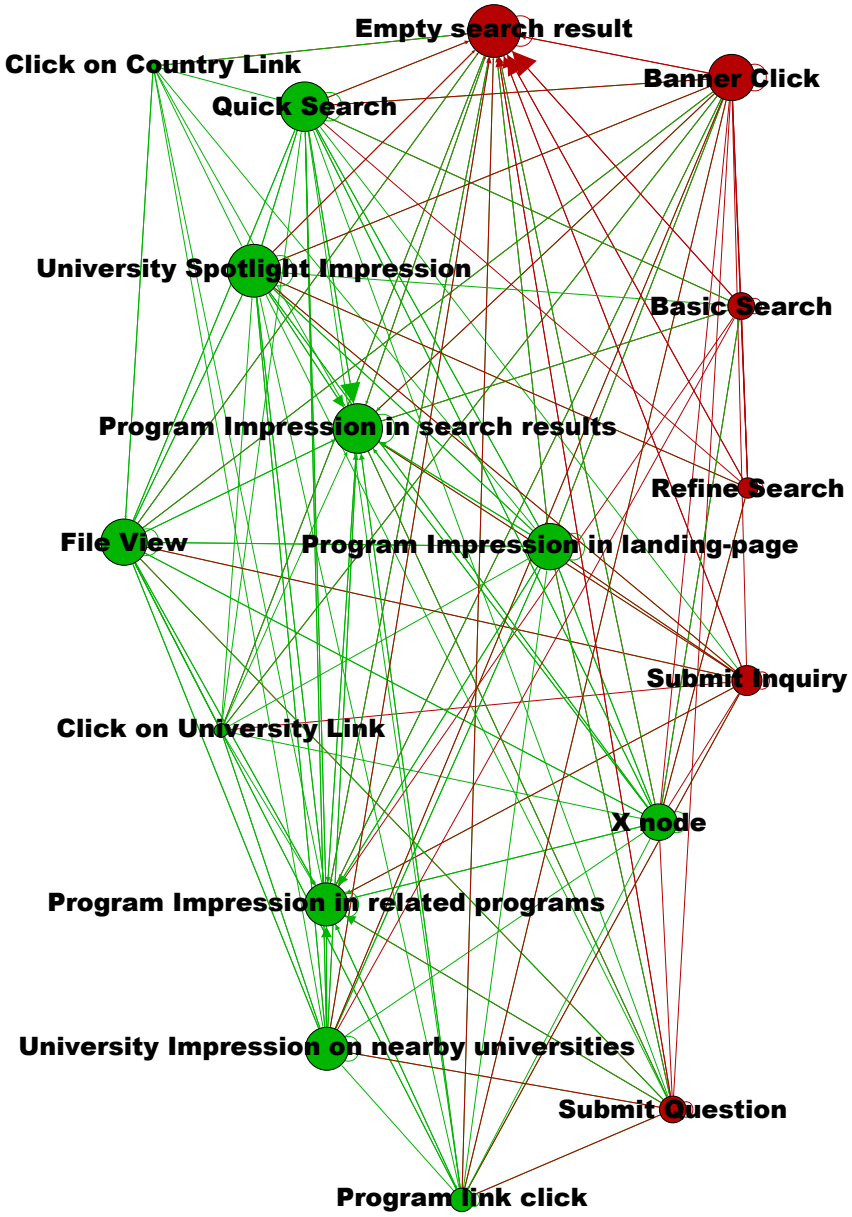



Figure 2.3: A user navigation graph. The meaning of nodes is described in Section 2.7.1 in detail. A graph partitioning algorithm is used to detect two communities in the graph: the red states are associated with 'expert' users and the green states are associated with 'novice' users.

to $V_{nov}$. Then, $A$ again sees another "novice" state $e$ and $S_i$ remains in $V_{nov}$. In the third step, $A$ encounters "expert" state $a$ and $S_i$ is moved to $V_{exp}$.

In this case, the type of user behavior is a context. The states in the navigation graph which are visited by a user during session $S_j$ is contextual features $\theta_{S_j}$. $\{V_i\}_{i=1}^{n}$ represent contextual categories. The proposed alignment approach allows us to effectively align training sequences to clusters. More importantly, the alignment is very convenient for sequentially aligning test sequences to clusters. In the experiments, we show that this approach works well with a specific real-world use-case. Moreover, the proposed approach can easily be generalized to any sequence data. Blondel et al. [35] introduce an algorithm that finds high modularity partitions of networks in a short time and that unfolds a complete hierarchical community structure for the network, thereby giving access to different resolutions of community detection. The modularity of a partition is a scalar value between -1 and 1 that measures the density of links inside communities as compared to links between communities. In case of weighted networks, it is defined as:

$$Q = \frac{1}{2m} \sum_{i,j} \left[ w_{i_j} - \frac{k_i k_j}{2m} \right] \delta(c_i, c_j), \tag{2.15}$$

where $w_{i_j}$ the weight of the edge between two vertices $i$ and $j$; $k_i = \sum_j w_{i_j}$ is the sum of the weights of the edges attached to vertex $i$, $c_i$ is the community to which vertex $i$ is assigned, the $\delta$ function is 1 if $u = v$ and 0 otherwise and $m = \frac{1}{2} \sum_{i,j} w_{i_j}$. The algorithm takes as input a weighted network of $N$ nodes. It is divided into two phases that are repeated iteratively. In the initial phase there are as many communities as there are nodes. Then, for each node $i$ it considers the neighbors $j$ of $i$ and the algorithm evaluates the gain of modularity that would take place by removing $i$ from its community and by placing it in the community of $j$. The node $i$ is then placed in the community for which this gain is maximum. This process is applied repeatedly and sequentially for all nodes until no further improvement can be achieved and the first phase is then complete. This first phase stops when a local maximum of the modularity is attained, i. e. when no individual move can improve the modularity. The second phase of the algorithm consists of building a new network whose nodes are now the communities found during the first phase. To do so, the weights of the links between the new nodes are given by the sum of the weights of the links between nodes in the corresponding two communities. Links between nodes of the same community lead to self-loops for this community in the new network. Once this second phase is completed, it is then possible to reapply the first phase of the algorithm to the resulting weighted network and to iterate.

The described algorithm is very simple, efficient, and perfectly fits into our schema to find groups of users' actions in the navigation graphs which is defined in definition 4. The resulted approach is summarized in Algorithm 1.

**Algorithm 1** User Expertise Discovery (UED)

1: **Input**: a sequence $S = s_1 s_2 \cdots s_k$
2: **Output**: sequence label as "expert" or "novice"
3: **for** $i \in \{1, 2, \cdots, k\}$ **do**
4:  **if** $(s_i \in V_{exp})$ **then**
5:  return "expert";
6:  **end if**
7: **end for**
8: return "novice"

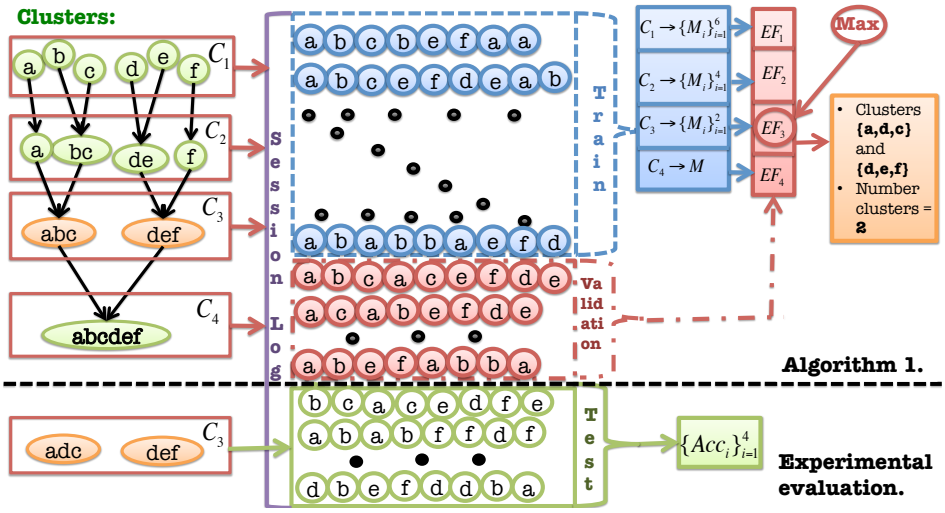



Figure 2.4: The general schema of proposed hierarchical clustering technique. The process of dividing training, validation and test sets is represented in Equation 2.20, Equation 2.22 and Equation 2.23 respectively.

### 2.6.3 Discovering User Intent Switch

We assume that the data are generated as follows: the events alphabet $A$ is produced by $h$ contexts. Under one specific context, the web session is generated by the events in that context only. Under that assumption, our goal is to decompose the web session (2.1), which is the sequence of users' actions on the site, into homogeneous pieces, such that the data in each segment can be described accurately by a temporal context $C_i$ and a simple prediction model $M_i$. Formally, we have to decompose the event space into clusters each corresponds to a hidden context.

The segmentation $Seg_i$ is defined separately for each web session $s_i \in D$ by $(k+1)$ segment boundaries: $1 = b_1 < b_2 < \cdots < b_k < b_{k+1} = (|s_i| + 1)$. The general segmentation for the set of sessions is specified as:

$$L = \{(Seg_1, \ldots, Seg_k)_i\}_{i=1}^{|D|}, \tag{2.16}$$

where $Seg_i$ is a homogeneous segment of the session. The problem is (1) to discover a set of $h$ contexts $\{C_{i=1}^h\}$ and (2) to decompose each $s_i \in D$ into $k$ segments and (3) for each segments $Seg_j$ and assignment of the context $C_i$.

**Definition 5** (Events Clustering)**.** *A clustering of web-session events $A$ is $E = \{E_1, \ldots, E_h\}$, where $E_i$ is a cluster of $A$ and each $E_i$ must satisfy the following properties:*

- *each cluster is not empty: $\forall E_i \in E : E_i \neq \emptyset$;*
- *all clusters are pairwise disjoint: $\forall i \neq j E_i \cap E_j = \emptyset$;*
- *every event belongs to one cluster: $\cup_i E_i = A$.*

The decomposition of the event space is not arbitrary but aims at maximizing the accuracy of the local models built for every context. Given the original log of sequences $D$ that is randomly divided two disjoint $D_{train}$, $D_{validation}$ and $D_{test}$. A decomposition of the event space into $h$ clusters uniquely splits each sequence in the data into segments. Let $T_{train} = \cup_{j=1}^{Freq(C_i)}(Seg_j \in D_{train})$ be the set of segments in the training set that corresponds the context $C_i$. We learn sets of predictive models $\{M_i\}_{i=1}^h$ based on $h$ sets of sessions segments. We validate our set of models based on $T_{validation} = \cup_{j=1}^{Freq(C_i)}(Seg_j \in D_{validation})$. And we test resulted clusters based on $T_{test} = \cup_{j=1}^{Freq(C_i)}(Seg_j \in D_{test})$. The effectiveness of one learner can be defined as:

$$F_{c_i}(T_{validation_i}, M_i) = \sum_{a_j \in T_{validation_i}} F(a_j, M_i) \tag{2.17}$$

For a given decomposition $E$, at the transition point, we always make wrong prediction. Therefore, the effectiveness of a decomposition $E$ can be defined as:

$$EF(E) = \sum_{i=1}^{h} F_{c_i}(T_{validation_i}, M_i) \tag{2.18}$$

The problem of the context switch discovery is defined as an optimization problem where the goal is to find the event space decomposition $E$ such that the prediction performance $EF(E)$ is maximized. We evaluate derived $E$ based on the test set $T_{test}$ and provide obtained accuracy that is calculate as in Equation 2.18.

Based on Equation 2.18 our objective function for the hierarchical clustering is:

$$EF^* = \arg\max_E EF(T_{validation}, M)), \tag{2.19}$$

where $E^*$ is optimal clustering that satisfies Definition 5.

Algorithm 2 shows the general framework of *Context Switch Discovery (CSD)*. We start by placing each item in its own cluster (line 1) and building predictive models based on training sets that are related to each cluster (line 2). For example, we have two clusters $E_i = \{a, b, c\}$ and $E_j = \{d, f, e\}$ and a training set $Train$. We split $Train$ into two parts that are related to each clusters accordingly:

$$\begin{aligned} Train = & \underbrace{abcb}_{\text{covered by } E_1} \; / \; \underbrace{ef}_{\text{covered by } E_2} \; / \; \underbrace{aa}_{\text{covered by } E_1} \\ & \underbrace{acb}_{\text{covered by } E_1} \; / \; \underbrace{efde}_{\text{covered by } E_2} \; / \; \underbrace{ab}_{\text{covered by } E_1} \\ & \underbrace{ababba}_{\text{covered by } E_1} \; / \; \underbrace{efd}_{\text{covered by } E_2} \end{aligned} \tag{2.20}$$

We obtain the following disjoint sets of sub-sessions:

$$\begin{aligned} Train_{E_1} &= \{abcb, aa, acb, ab, ababba\} \\ Train_{E_2} &= \{ef, efde, ef, efd\} \end{aligned} \tag{2.21}$$

Next, we build two prediction models: $M_1$ and $M_2$ based on $Train_{E_1}$ and $Train_{E_1}$ respectively. Then, we calculate effectiveness of obtained models by applying them on $Validation$ (line 3). We decompose the validation session using the same way as in 2.20:

$$\begin{aligned} Validation = & \underbrace{abcac}_{\text{predict by } M_1} \; / \; \underbrace{efde}_{\text{predict by } M_2} \\ & \underbrace{acab}_{\text{predict by } M_1} \; / \; \underbrace{efde}_{\text{predict by } M_2} \\ & \underbrace{ab}_{\text{predict by } M_1} \; / \; \underbrace{ef}_{\text{predict by } M_2} \; / \; \underbrace{abba}_{\text{predict by } M_1} \end{aligned} \tag{2.22}$$

Then we iteratively detect two clusters which increase the general effectiveness (line 7) of the predictive system and merge them (line 9). In other words, during each iteration the method tries to increase the general effectiveness (2.18) of the system. If a merge increases the system effectiveness, Algorithm 2 keeps it (line

**Algorithm 2** Context Switch Discovery (CSD)

**Input:** A web-session dataset $D$, train sessions $Train$, validation sessions $Validation$, test sessions $Test$, predefined type of predictive model $M$
**Output:** A clustering of events $E = \cup_{i=1}^{h} E_i$

1: $E \leftarrow \{\{a\} | a \in A\}$
2: $M \leftarrow buildPredictiveModels(Train, E))$
3: $EF_{opt} \leftarrow \text{argmax}_E \, E(Validation, M)$
4: **while** $|E| > 1$ **do**
5: $E_i, E_j$
6: $M_k \leftarrow buildPredictiveModel(Train \cap (E_i \cup E_j))$
7: **if** $EF(Validation, M_k) > EF_{opt}$ **then**
8: $EF_{opt} \leftarrow EF(Validation, M_k)$
9: $E \leftarrow E_i \cup E_j$
10: **end if**
11: **end while**
12: **return** $E$

8-9). We have as a result a set of *clusters* $E$ and a set of predictive models $\{M_k\}_{k=1}^{h}$. We return *best clusters* (line 12).

The additional benefit of the suggested method is that it produces the "best" number of clusters thus discover the number of hidden contexts. The general schema of proposed clustering technique is presented also in Figure 2.4 for our the running example.

**Evaluation of hierarchical clustering** In order to evaluate the obtained clusters we decompose the test set $Test$ according to result $E = \{E_1, E_2\}$ as presented in Equation 2.23. As an evaluation metric we use accuracy which is presented in Equation 2.24.

$$Test = \begin{array}{ccccc} \underbrace{bcac}_{\text{predict by } M_1} & / & \underbrace{edfe}_{\text{predict by } M_2} & & \\ \underbrace{abab}_{\text{predict by } M_1} & / & \underbrace{ffdf}_{\text{predict by } M_2} & & \\ \underbrace{ab}_{\text{predict by } M_1} & / & \underbrace{fdd}_{\text{predict by } M_2} & / & \underbrace{ba}_{\text{predict by } M_1} \end{array} \tag{2.23}$$

$$Accuracy(E) = \sum_{i=1}^{h} F_{c_i}(T_{test_i}, M_i) \tag{2.24}$$

The general schema of the proposed experimental evaluation is described also in Figure 2.4 (Experimental Evaluation) for either our or the running example.

## 2.7 Experimental study

This section outlines the experimental methodology we employed to answer our **RQ 2.6** *How effective is our approach to discover useful contexts on a realistic sample of traffic?* in the next section. The goal of experimental study is to show that introducing the list of useful contexts improves predictive accuracy of the model. First, we describe the dataset that we use for experiments (Section 2.7.1). Then, we present our methodology for the experiment design (Section 2.7.2).

### 2.7.1 Data

The anonymized dataset for our case study comes from StudyPortals. The web-portal provides information about various study programmes in Europe. We used data that was collected in May 2012, the dataset contained over 350.000 sessions[1]. The revenue model for the StudyPortals is based on selling web-based advertising campaigns and providing sponsored search-results. Thus, it is vital to increase both the number of visitors and the number of pages viewed by each visitor, but ultimately to help visitors to find relevant information and enrol into a study program they are interested in.

Each user's session is recorded as following:

- *user identifier (based on IP address and cookie information);*
- *timestamp for start of the user session;*
- *timestamp for end of the user session; and*
- *the sequence of the user's actions during the session.*

StudyPortals has a categorization of actions that users can perform on the website. This categorization is used to describe users' paths on the website which can be transferred into a navigation graph. The users' navigation for the Study-Portals is demonstrated in Figure 2.3, where possible user actions $a_i$ are presented: $A = \{a_1, a_2, \ldots, a_{16}\}$. General types of action on the site are:

- **view** (exp. **view** study file with additional information);
- **click** (**click** on a banner, or a country information link, or a university link, or a program link);
- **submission** (user feedback through question or inquiry **submit**);
- **impression** (referee to the recommendation actions);
- **search** (**quick search** that is a simple search from the homepage);
- **basic search** (when a user uses special search page);

[1]The dataset is publicly available as a benchmark, for details we refer to `http://www.win.tue.nl/~mpechen/projects/capa/#Datasets`

- **refined search** (when additional filters are used); and
- **X node** (other actions which are out of the categorization scope).

### 2.7.2 Experiment Design

Our sequential dataset of users' actions is split randomly into two parts: test set contains $T$ - 20% of the session log and training set ($Tr$) contains - 80% of that log.

**Training phase.** The whole $Tr$ is used to learn a global predictive model $M$ (Glob.). Using the *sequential user alignment* method we divide the whole training dataset into subsets, which apply to each of the contexts. Assume we have context $C$ with $k$ categories $\{c\}_{i=1}^{k}$ dividing the train set into $k$ disjoint subsets $\{Tr_i\}_{i=1}^{k}$ such that $Tr = Tr_1 \cup Tr_2 \dots Tr_k$. Let denote $\{M_i\}_{i=1}^{k}$ as $k$ predictive models built for categories $\{c_i\}_{i=1}^{k}$ respectively.

**Testing phase.** During the testing stage we calculate the accuracy of global model $M$ - $Acc[T, M]$ (Equation 2.6). $k$ categories $\{c\}_{i=1}^{k}$ divide the test set $T$ into $k$ disjoint subsets $\{T_i\}_{i=1}^{k}$ such that $T = T_1 \cup T_2 \dots T_k$. Let $P(c_i)$ be the probability that the test instance belongs to the category $c_i$ then $Acc[T, \{M\}_{i=1}^{k}] = \sum_{i=1}^{k} P(c_k) Acc[T_i, M_i]$ (Equation 2.7).

We experiment with various of predictive Markov models $M$ introduced in Section 2.4. We use the following Markov models: first order Markov models (FOMM) and variable order Markov models: Context Tree Weighting (CTW) [243] and Probabilistic Suffix Trees (PST) [192]. To calculate the final metrics we run the described evaluation procedures 10 times to collect average metrics. We run the evaluation cycle for two discussed contexts: users' type ("novice" vs. "expert") and geographical location. The contextual models are compared agaist global non-contextual predictive model.

## 2.8 Results and Findings

In this section we answer our **RQ 2.6** *How effective is our approach to discover useful contexts on a realistic sample of traffic?* We start by presenting the results for the explicit context based on a user's geographical location and how it affects the overall quality of prediction (Section 2.8.1). Then, we focus on discovered context based on type of user's behavior, also we present results for the random context (Section 2.8.2). We present our final results by reporting the relative improvements over the predictive accuracy. We investigate the robustness of our finding by suggesting several constraints for predictions. Finally, we present results where we discover user intent switch within a web session (Section 2.8.3).

### 2.8.1 Geographical Location as Context

In this experiment we want to evaluate an impact of context based on geographical location of a user. We use mapping from contextual feature *IP address* to a

Table 2.1: Average accuracies (± standard deviation) of user intent prediction with the global Markov and local ("location" context) Markov models. "Glob." - global model accuracy, "W.Sum" - weighted sum of local model accuracies (Equation 2.6), "RI" - relative improvement compared to the global models.

| Cat. $c_i$ | Size $c_i$ | FOMM(%) | CTW(%) | PST(%) |
|---|---|---|---|---|
| Global | 1 | 40.6±0.3 | 49.2±4.3 | 45.3±0.2 |
| EU | 0.45 | 45.0±0.4 | 48.3±4.4 | 47.3± 3.9 |
| AS | 0.27 | 38.9±0.4 | 47.4±4.1 | 44.4± 3.3 |
| AF | 0.08 | 34.4±0.7 | 48.5±3.2 | 48.4± 3.2 |
| NA | 0.16 | 35.8±0.8 | 48.3±5.2 | 49.1± 4.9 |
| SA | 0.02 | 41.7±1.7 | 48.1±1.6 | 50.2±4.1 |
| OC | 0.01 | 46.8±2.8 | 45.2±6.4 | 49.4±9.1 |
| W.Sum | 1 | 40.1±0.4 | 48.3±2.6 | 46.1±1.4 |
| RI | - | **-1.2** | **-1.8** | **+1.8** |

continent as alignment method to cluster the session. Therefore, we have six contextual categories: *EU* - users from continent Europe, *AS* - users from continent Asia, *AF* - users from continent Africa, *NA* - users from continent North America, *SA* - users from continent South America, *OC* - users from continent Oceania. We derive six separate predictive models for each continent: $\{M_{c_i}\}_{i=1}^{6}$ that is trained for each continent and one global prediction model $M$ that is trained on whole dataset as output of training stage.

The resulting accuracy is shown in Table 2.1. Clearly, the user's geographical location is not a useful context according *Definition* 3. Because the way as the context divides data does not give us any benefits in terms of $Acc[T, \{M_i\}_{i=1}^{6}]$ that is accuracy. The related improvements of the predictive accuracy are almost always negative. Only for the case of PST location context gives slightly improvement. Therefore, the geographical location is not a useful context for our domain in this particular use-case.

### 2.8.2 User Expertise as Context

In this experiment we want to evaluate an impact of the context based on discovered communities in the user navigation graph. The method to obtain the context based on the type of user behavior is described in detail in Section 2.6.2. By applying this method we obtain two communities in our users' navigation graph with modularity equals to 0.174.

$M$ is a global model that is built on the whole $Tr$. We use the alignment method and as a result we obtain two clusters: $V_{exper}$ and $V_{novice}$. These clusters are used to learn two contextual models: $M_{expert}$ and $M_{novice}$. The resulting accuracy is shown in Table 2.2. The related improvement compared to the performance of the global model is high, up to 18.9% in terms of the PST predictor. Distinctly, the type of user behavior is a useful context according to *Definition* 3.

Table 2.2: Average accuracies (± standard deviation) of user intent prediction with the global Markov and local ("user type" context) Markov models. Relative improvement compared to the global model ("Glob.") is given in bold in the round brackets. "W.Sum" is weighted sum of the local model accuracies (Equation 2.6).

| Cat. $c_i$ | Size $c_i$ | FOMM(%) | CTW (%) | PST (%) |
|---|---|---|---|---|
| Global | 1 | 40.6±0.3 | 49.2±4.3 | 45.3±0.2 |
| "expert" | 0.11 | 55.3±0.9 **(+36.2)** | 59.3±3.1 **(+20.5)** | 60.7±1.8 **(+34.0)** |
| "novice" | 0.89 | 43.4±0.3 **(+6.9)** | 53.2±1.9 **(+8.3)** | 53.1±2.9 **(+17.2)** |
| W.Sum | 1 | 43.4±0.28 **(+6.9)** | 54.4±1.7 **(+10.6)** | 53.9±2.7 **(+18.9)** |

Since this context gives improvement in terms of $Acc[T, M_1, M_2]$, for all given predicting models: for FOMM the relative improvement is 6.9%, for CTW the relative improvement is 10.6%, and for PST the relative improvement is 18.9%. According to the predictive accuracy it is important to notice that we have much higher relative improvement for the "advanced" users which are our target group from our business perspective. This group of users has longer sessions which again indicates their interest to find a suitable program. Therefore the type of user behavior is a useful context for our domain in particular use-case of users' trail prediction.

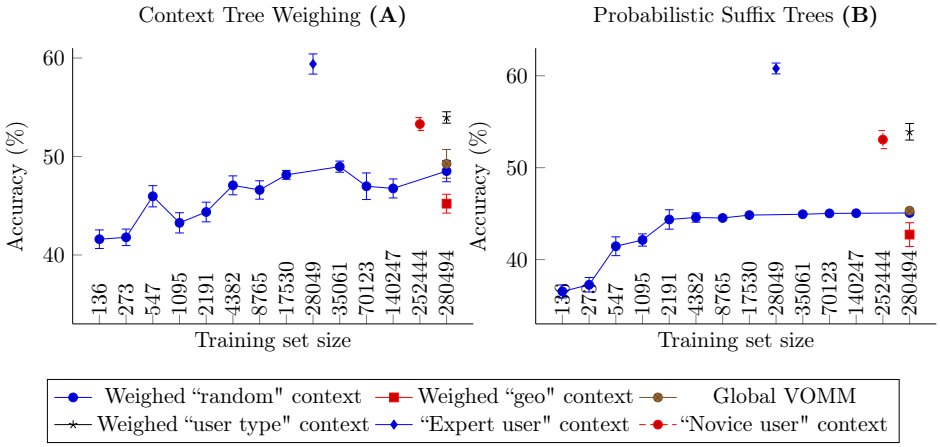
Therefore, the proposed technique to discover *useful contexts* that can be used to improve the prediction models for the user navigation *trails*.

**Random Context** We introduce a random context $R$ in order to provide a support evidence for the presented theory about contextual Markov models. In particular, we aim to provide an experimental argument that local Markov models are not worse than global Models.

We randomly select training samples of different size. Assume that a random context has two categories, so $k = 2$. Therefore, we divide $Tr$ randomly into two samples $(Tr/2)_1$ and $(Tr/2)_2$ and build local models $M_{Tr/2_1}$ and $M_{Tr/2_2}$ respectively. An alignment function randomly selects model $M_{Tr/n_i}$ for a test instance. Then the expected accuracy $Acc[T, M_{Tr/2_1}, M_{Tr/2_2}]$ is calculated (Equation 2.7). We continue the experiment, recursively splitting the training data until the size of $Tr/n$ becomes less than 100 sessions. We run the experiment 10 times and compute averages and standard deviations of generalization accuracies.

The results are presented in Figure 2.5. The blue plot "Weighed random context" shows the accuracy of local models of different size. Figure 2.5 (B) presents the results for the PTS predictor. We can clearly see that when the training size becomes less than 4k the standard error increases substantially and the accuracy declines. The same situation occurs with the CTW predictor in Figure 2.5 (A) - accuracy drops when the training size becomes less than 4k instances. Both predictors show the same tendency - the accuracy decreases when the size of the

Figure 2.5: Mean of accuracy for 10 iterations with standard error (SE). Plot (A) represents results for the CTW algorithm. Plot (B) represents the results for the PST algorithm.




sampled training subset is less than 10-20% of the whole set, and an increase of the standard error testifies about future reduction of accuracy (or future unexpected behavior). Figure 2.5 also depicts accuracies and the corresponding standard errors of the global model and considered contexts: geographical location and users' behavior type. Based on the observations we can hypothesize that if the standard error is low then the discovered cluster is strong.

### 2.8.3 Intent Switch as Context

**Baseline for Intent Switch Discovery** An alternative approach to analyze the sequential data is a Hidden Markov Model (HMM) [111]. In order to obtain clusters of user activities we have implemented *Context Switch Discovery (CSD)* that is presented in Algorithm 2. Each cluster is associated with a context according to Definition 2. We have run both algorithms (HMM and CSD) 10 times. The method was trained and tested based on randomly split data. The obtained average metrics are presented in Figure 2.6. It is clear from the plot that CSD outperforms HMM. The accuracy of global model "Global M." is depicted as a green dot in Figure 2.6. The global model equals to CSD when CSD has one cluster. We also show that the set of contextual local models outperforms the global model.

Our objective function for CSD reaches a maximum when the number of clusters is 7. Therefore, an average number of elements in the cluster is 2.3. This happens because users use repetitive states as shown in Equation 2.25.

$$s_i = \langle \underbrace{(A_1, t_1), \ldots, (A_i, t_i), \ldots, (A_8, t_8)}_{A_i = \text{'Program impressions in related programs'}} , \ldots , \underbrace{(A_9, t_9)}_{A_9 = \text{'Click on Banner'}} \rangle \quad (2.25)$$

Figure 2.6: Resulted effectiveness

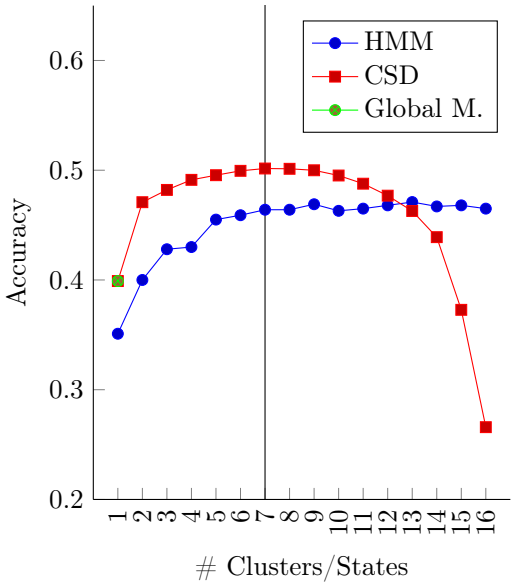



Table 2.3: Resulting clusters

| Id | Summary | Cluster |
|---|---|---|
| 1 | Intensive Search | Basic Search, Refine Search, Empty Search Result |
| 2 | Explore information related to program | Program impression in search result, Banner click, Program click , Click on university link |
| 3 | Start of browsing | University Spotlight impression, Quick search |
| 4 | Explore country information | File view, Click on country link |
| 5 | Explore search result | Program impressions in search results, University impression on nearby universities |
| 6 | Explore program | Program in landing page, Submit inquiry |
| 7 | - | Submit question, X-node |

Table 2.3 presents seven clusters that we discovered in the dataset. We provide a summary for each cluster in the table. Such summary can be used to provide a user's roadmap on the site. It helps to better understand the user's behavior on the site and to discover existing problems. Like in our example, the cluster "Intensive Search" shows that users encounter a problem in order to find information. The cluster 7 looks like an outlier because the action 'Submit question' can be made from every page on the site, and we can't decipher 'X-node'. It is a good point that the algorithm can filter out the noise.

Figure 2.7 provides some further insights on the interpretability of the obtained clustering. The figure shows the correspondence of available navigation across the

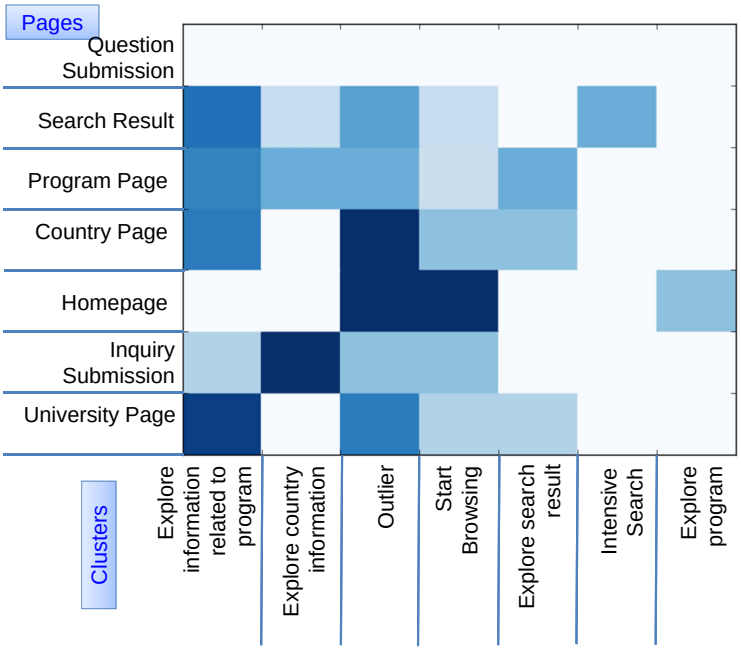



Figure 2.7: Visualization of the clusters.

website pages and obtained clusters. The intensity of colors corresponds to the values of Jaccard index between the page and the cluster. There is a strong relation between the possible actions that users can perform at certain website pages and corresponding clusters; there are five pages that correspond pretty well to four clusters. We are not interested in the cluster 'Outlier', so we do not discuss it:

1. 'University page' corresponds to cluster 'Explore information related to program';
2. 'Inquiry submission page' corresponds to cluster 'Explore country information';
3. 'Homepage' corresponds corresponds to cluster 'Start Browsing'.

This type of analysis might help to improve the design of a website. But for our study it serves as additional evidence that the proposed approach for hidden context discovery is reasonable.

## 2.9 Conclusions

This chapter investigates how identify contexts that help us to predict users' trails, which is an important task of behavioral targeting. In practice, domain experts

can have many ideas about possible contexts for the domain, based on their intuition, or context is identified using data mining techniques. However, we need to justify whether the overall system would benefit from the proposed context. It will be important to be able to detect the effectiveness in an offline evaluation because the online experimentation can be expensive and time consuming, especially for web services without a massive amount of traffic. Our main research question in this chapter was: *How to automatically discover a set of useful contexts to predict a user's next action in a web session?*

We introduced a formal definition of useful context and defined the contextual principle to answer **RQ 2.1** *What is contextual prediction?* We formulated the context discovery as a straightforward optimization problem.

We used Markov models to approach **RQ 2.2** *What machine learning methods can be used to discover useful context for the next action prediction?* We provided intuitive proofs showing that an optimal global model corresponds to optional contextual models, and that for Markov models the contextual models are expected to be at least as good as global ones. We performed an experiment with random contextual Markov models and it shows that with some constraints they are almost as good as global models. This fact gives us experimental evidence about the robustness of the proposed contextual models. Thus, at least for this class of models, we have a sound justification and motivation for context-aware predictive analytics. We introduced a method for context discovery which consists of two important components: a clustering algorithm which divides training sequences into $k$ groups and an alignment method which assigns each new test sequence to the most appropriate cluster. We presented experiments with real-world dataset for two specific examples of our method: (1) explicit contexts of users' location which is widely used in many applications and (2) implicit context that is inferred from users' navigation graphs.

To investigate **RQ 2.3** *What is the impact of geographical location as contextual information?* we used geographical location on a continent level for our contextual prediction models, as there was not enough data to support modeling locations at a finer grained level. As it turned out, the geographical location is not a useful context according our earlier defined contextual principle. A possible explanation is that the general audience of StudyPortals consists of a relatively homogeneous group of students with a similar level of education, approximately the same age and having a high level of proficiency in English.

To answer **RQ 2.4** *How to discover types of user behavior based on performed actions?* we discovered two types of user behavior on the site by grouping user based on their navigation graph: novice and expert users. This discovered contextual information improved the prediction accuracy.

To investigate **RQ 2.5** *How to discover changes of user intent within a web session?* we applied contextual principle to discover changes in user intent during a single session. We showed that the prediction accuracy can be improved by taking into account the context switches that may occur, leading to more accurate models of the user's intent.

Finally, to answer **RQ 2.6** *How effective is our approach to discover useful contexts on a realistic sample of traffic?*we performed an experimental case study

on real world data that can be regarded as an illustration of contextual Markov models learning. This case study shows that if we can identify useful contexts, the local Markov models outperform the single global Markov, and if context is not useful, local models will still perform as good as the global model.

# 3
# Contextual User Profiles

Many e-commerce websites use recommender systems or personalized rankers to personalize search results based on their previous interactions. However, a large fraction of users has no prior interactions, making it impossible to use collaborative filtering or rely on user history for personalization. Even the most active users may visit only a few times a year and may have volatile needs or different personas, making their personal history a sparse and noisy signal at best. This chapter investigates how, when we cannot rely on the user history, the large scale availability of other user interactions still allows us to build meaningful profiles from the contextual data and whether such contextual profiles are useful to customize the ranking, exemplified by data from a major online travel agent.

Our main findings are threefold: First, we characterize the *Continuous Cold Start Problem* (*CoCoS*) from the viewpoint of typical e-commerce applications. Second, as explicit situational context is not available in typical real world applications, implicit cues from transaction logs used at scale can capture essential features of situational context. Third, contextual user profiles can be created offline, resulting in a set of smaller models compared to a single huge non-contextual model, making contextual ranking available with negligible CPU and memory footprint. Finally we conclude that, in an online A/B test on live users, our contextual ranker increased user engagement substantially over a non-contextual baseline, with click-through-rate (CTR) increased by 20%. This clearly demonstrates the value of contextual user profiles in a real world application.

## 3.1 Introduction

In addition to the handful of general web search engines, there are millions of online e-commerce websites driving the online economy [45]. Many of these e-commerce websites are built around personalized search and recommendations systems. Amazon.com recommends books, Booking.com recommends accommodations and destinations, Netflix recommends movies, Reddit recommends news stories and so on. Recommender systems predict unknown ratings based on past

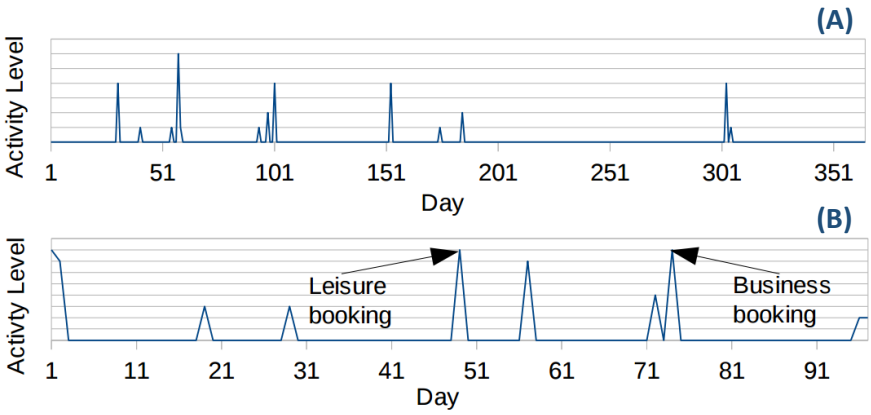



Figure 3.1: Continuously 'cold' users at Booking.com. Activity levels of two randomly chosen users over time. (A): The top user has only rare activity throughout a year. (B): the bottom user exhibits different personas by making a leisure and a business booking without much activity in between.

or/and current information about users and items, such as past user ratings, user profiles, item descriptions. If this information is not available for new users or items, the recommender system runs into the *Standard Cold Start Problem*: it does not know what to recommend until the new, 'cold' user or item gets 'warmed-up', i. e. until enough information has been received to produce recommendations. For example, which hotels should be recommended to someone who visits Booking.com for the first time? If the recommender system is based on the history of users click' in the past, the first recommendations can only be made after the user has clicked on a couple of hotels on the website.

Several approaches have been proposed to deal with the cold-start problem, such as utilizing baselines for cold users [154], combining collaborative filtering with content-based recommenders in hybrid systems [201], eliciting ratings from new users [187], promoting diversity in recommendations [105], or exploiting the social network of users [204]. In particular, content-based approaches have been very successful in dealing with cold-start problems in collaborative filtering [200, 201]. However, these approaches deal explicitly with 'cold' users or items, and provide a 'fix' until enough information has been gathered to apply the core recommender system. Thus, rather than providing unified recommendations for 'cold' and 'warm' users, they temporarily bridge the period during which the user or item is 'cold' until it is 'warm'. This can be very successful in situations in which this warm-up period is short, and when warmed-up users or items stay warm.

However, in many practical e-commerce applications, users or items remain 'cold' for a long time, and can even 'cool down' again, leading to the *Continuous Cold Start Problem* (*CoCoS*). For example in Booking.com, many users visit and

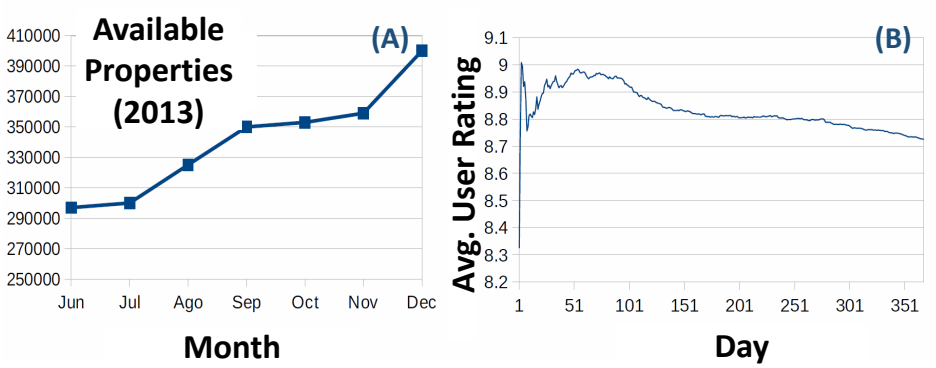



Figure 3.2: Continuously cold items at Booking.com. (A): Thousands of new accommodations are added every month. (B): The user ratings of a randomly chosen hotel change continuously over the year.

book infrequently because they have only one or two vacations per year, leading to a prolonged cold-start and extreme sparsity of collaborative filtering matrices, see Figure 3.1 (A). In addition, even 'long term warm' users can cool down as they change their needs over time [129], e. g. coming from Booking.com of youth hostels for backpacking to booking of resorts for family vacations. Such 'cool-downs' can happen more frequently and rapidly for users who book accommodations for different travel purposes, e. g. for leisure holidays and business trips as shown in Figure 3.1 (B). Moreover, we have a mirror problem in the items to recommend: new items appear frequently leading to many items without prior interactions as shown in Figure 3.2 (A) for accommodations at Booking.com, and items can change their characteristics as shown in Figure 3.2 (B), making historical interactions a noisy signal. The *CoCoS* is ignored in the literature despite its relevance in industrial applications. Classical approaches to the cold-start problem fail in the case of a *CoCoS*, since they assume that users get warmed up in a reasonable time and stay warm after that.

This chapter proposes a new approach of using contextual user profiles for personalized search and recommendations in the context of a major online travel agent, in particular using the Destination Finder. Situational context provides powerful cues about user preferences that hold the promise to improve the quality of recommendations over the use of traditional long term interests [e. g., 5, 21, 22]. In this setup, rankings are computed based on the current context of the current visitor and the behavior of other users in similar contexts [e. g., 6, 100, 208]. This type of data is readily available in most e-commerce settings. This approach naturally addresses sparsity by clustering users into contexts. Since context is determined on a per-action basis, user volatility and multiple personas can be addressed robustly.

Working in a real world setting comes with specific challenges for search and recommendation systems [155]. First, in an online service, context is shallow but

available at scale. Context can be almost anything—ranging from explicit user profiles to data about moods and attitudes—but explicit user context is typically not available in online services. There is an abundance of situational context (day, time, device, etc) in server logs which may hold important implicit contextual cues. Hence, although rich contextual information is not available for a large fraction of users, the large scale availability of implicit situational context may still allow us to capture essential context features. Second, if it's not fast it isn't working. Due to the volume of traffic, offline processing—done once for all users—comes at marginal costs, but online processing—done separately for each user—can be excessively expensive. Clearly, response times have to be sub-second, but even doubling the CPU or memory footprint comes at massive costs. Hence we cannot include implicit contextual features directly or build an adaptive model for each unique user, but we can build profiles offline and map incoming users to one of the profiles at negligible online processing costs.

We are trying to answer the following main research question: *Can we automatically detect contextual user profiles and does customized ranking with these profiles improve travel search and recommendation?* We break down our general research problem into four specific research questions:

> **RQ 3.1:** *How to characterize the continuous cold start problem in travel recommendation?*

We introduce and characterize the *Continuous Cold Start Problem* (*CoCoS*) that happens when users or items remain 'cold' for a long time, and can even 'cool down' again after some time.

> **RQ 3.2:** *How to define and discover contextual user profiles from multi-criteria ranking data in an unsupervised setup?*

We combine multi-criteria ranking data with the $n$-dimensional contextual space in order to discover contextual user profiles.

> **RQ 3.3:** *How to apply contextual user profiles for the ranking of travel destinations in a continuous cold start setting?*

We propose a novel approach exploiting contextual user profiles which are defined as *'closely connected'* regions of an $n$-dimensional contextual space.

> **RQ 3.4:** *How effective are contextual profiles for real-world users of the destination finder system in terms of user engagement measures?*

We set up a large-scale online A/B testing evaluation with live traffic from Booking.com, and demonstrate how contextual travel ranking leads to a significant increase in user engagement.

The remainder of this chapter is organised as follows. In Section 3.2 we discuss the most relevant prior work, and position our research with respect to it. The problem setup is introduced in Section 3.3. As our approach is generally applicable to any multi-criteria ranking data associated with standard contextual

information from web logs, Section 3.4 outlines our approach as a general framework for discovering and using contextual user profiles. Next, in Section 3.5, we detail the specific application to our online travel agent service. In Section 3.6, we describe the results of the online evaluation of the approach in an A/B test with live traffic. Finally, Section 3.7 concludes our work in this chapter and highlights future directions.

## 3.2 Background and Related Work

In this section, we review related work in the following two areas. First, we summarize previous work on the attempts to solve *CoCoS*. Second, we review approaches to build situational recommendations.

### 3.2.1 Cold Start Problem

In classical formulations of Recommender Systems (RS), the recommendation problem relies on *ratings* ($R$) as a mechanism of capturing user ($U$) preferences for different items ($I$). The problem of estimating unknown ratings is formalized as follows: $F : U \times I \rightarrow R$. Due to practical applications, RS have been an expanding research area since the first papers on collaborative filtering in the 1990s [190, 205]. Many different recommendation approaches have been developed since then, in particular content-based and hybrid approaches have supplemented the original collaborative approaches [3]. For instance, RS based on latent factor models have been effectively used to understand user interests and predict future actions [7, 8]. Such models work by projecting users and items into a lower-dimensional space, thereby grouping similar users and items together and subsequently computing similarities between them. This approach can run into data sparsity problems and into *CoCoS* when new items continuously appear. Although, to our knowledge, the *CoCoS* as defined in this work has not been directly addressed in the literature, several approaches are promising.

Tang et al. [225] propose a context-aware recommender system, implemented as a contextual multi-armed bandits problem. Although the authors report extensive offline evaluation (log based and simulation based) with acceptable CTR, no comparison is made from a cold-start problem standpoint.

Sun et al. [222] explicitly attack the user volatility problem. They propose a dynamic extension of matrix factorization where the user latent space is modeled by a state space model fitted by a Kalman filter. Generative data presenting user preference transitions is used for evaluation. Improvements of RMSE when compared to time SVD [156] are reported. Consistent results are reported in [52], after offline evaluation using real data.

Tavakol and Brefeld [226] propose a topic driven recommender system. At the user session level, the user intent is modeled as a topic distribution over all the possible item attributes. As the user interacts with the system, the user intent is predicted and recommendations are computed using the corresponding topic distribution. The topic prediction is solved by factored Markov decision processes.

Evaluation on an e-commerce data set shows improvements when compared to collaborative filtering methods in terms of average rank.

### 3.2.2 Context-Aware Recommendations

The radical departure from classical, two-dimensional RS is context-aware recommendation system (CARS) [4], which attract an increasing attention in academic work [91, 92, 208]. Rating prediction in CARS relies primarily on the information of *how* (which rating, e. g. a user giving '3' of '5' stars to an item) and *who* (which user, e. g. gender, mood or nationality) rated *what* (which item, e. g. movie, news article, or hotel). This additional information is called *context*. The general formulation of CARS rating prediction takes into account the context dimension $C$ as follows [4]:

$$F : U \times I \times C \to R. \tag{3.1}$$

Defining context is an important research question in itself. The structured definition of context was introduced in [44]. *Multidimensional context* $C$ is defined as a group of contextual feature-category pairs:

$$C = \{(F_n : \{v_m\}_{m=1}^{M})\}_{n=1}^{N}, \tag{3.2}$$

where $F_n$ are contextual features, and $v_m$ are categories for $F_n$. For example, the contextual feature *location* has the contextual categories 'USA', the 'Netherlands' etc. *Contextual categories* are often predefined by taxonomies [24, 100, 251]. Alternatively, an unsupervised technique is used to discover contextual information [144, 174]. Moreover, context discovery can be formulated as an optimization problem [143] or a feature selection problem [233, 235].

Incorporating contextual information into CARS can be viewed as a separate area of research, and can be classified into five groups [4] presented in Section 1.1.4. Adomavicius et al. [5] introduce a multidimensional approach taking various contextual aspects into account in collaborative filtering. They use a reduction based approach mapping a three-dimensional prediction function (of Equation 3.2) to a two-dimensional one. Baltrunas and Ricci [21, 22] introduce item splitting for dealing with context by generating new items, where context sensitive items are duplicated and the ratings divided over the respective contextual conditions, reducing it to a classical RS problem. This approach is expanded by Baltrunas and Ricci [23] and evaluated on synthetic and real world data sets.

Contextual information is initially ignored for post-filtering approaches, which also can be referred to as contextualization of the recommendation output [181]. The ratings are predicted using any traditional two dimensional RS set-up on the entire data. Then, the resulting set of recommendations is adjusted (contextualized) for each user using the contextual information.

A common context modeling approach is to use contextual information to expand the feature set, thus treating context as a predictive feature. For example, Rendle et al. [189] proposed a novel approach applying Factorization Machines [188] to model contextual information and provide context-aware rating

predictions, *using context explicitly* specified by a user to expand *the set of predictive features*.

Tensor Factorization, which is a generalization of Matrix Factorization, allows a flexible and generic integration of contextual information by modeling the data as a User-Item-Context $N$-dimensional tensor instead of the traditional 2D RS [130, 207]. In terms of an interactive system, Palmisano et al. [180] has shown that it was useful to consider the history of user interactions, more specifically changes in these entities. In [91], a co-occurrence analysis is used to mine the top frequent tags for songs from social tagging web sites, and topic modelling is used to determine a set of latent topics for each song. Recently, more techniques for context modeling were developed [27, 92, 224].

In multi-criteria RS [2, 6, 160] (MCRS) the rating function has the following form:

$$F : U \times I \rightarrow r_0 \times r_1 \cdots \times r_n. \tag{3.3}$$

The *overall rating* $r_0$ for an item shows how well the user likes this item, while criteria ratings $r_1, \ldots, r_n$ provide more insight and explain which aspects of the item she likes. MCRS predicts the overall rating for an item based on the past ratings, using both overall and individual criteria ratings, and recommends to users the item with the best overall score. According to [2], there are two basic approaches to compute the final rating prediction in the case when the overall rating is known. First, in *similarity based* approaches, the similarity between users is calculated based on their detailed ratings (e. g. Euclidean distance, Chebyshev distance, or Pearson correlation). Second, in *aggregation function based* approaches, we exploit the assumption of a relationship between the overall and the criteria ratings, $r_0 = f(r_1, \ldots, r_k)$ (e. g. multiple linear regression techniques can be used). These two approaches have been significantly improved in [117] by using Support Vector regression and combining user- and item-based regression models with a weighted approach. Liu et al. [170] assumed that the overall rating highly correlates with criteria ratings that are particularly significant for individuals.

RS methods are not easy to apply for large scale industrial applications. A large scale application of an unsupervised RS is presented in [106], where the authors apply topic modeling techniques to discover user preferences for items in an online store. They apply Locality Sensitive Hashing techniques to overcome performance issues when computing recommendations.

To summarize, the key distinction of our work compared to previous efforts is twofold: First, we introduce new Continuous Cold Start (*CoCoS*) settings that is common in e-commerce. Second, we propose the discovery of contextual user profiles (CUPs) within a *CoCoS* setting. CUPs are used both to build customized context-aware rankers (which can be done offline), and to map incoming users to the closest contextual user profile to provide contextual recommendations.

## 3.3 Problem Setup

In this section we will study our ***RQ 3.1:*** *How to characterize the continuous cold start problem in travel recommendation?* First, we characterize the *Contin-*

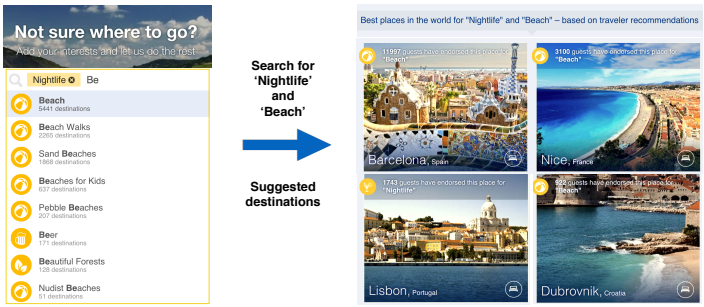



Figure 3.3: Example of Destination Finder use: a user searching for 'Nightlife' and 'Beach' obtains a ranked list of recommended destinations (top 4 are shown).

*ues Cold Start Problem* (*CoCoS*) in Section 3.3.1. Second, we introduce a Booking.com service Destination Finder that 'suffers' from *CoCoS* in Section 3.3.2. It will be our platform for experimentation in the remainder of the chapter.

### 3.3.1 Characterizing Continuous Cold Start

*CoCoS* can in principle arise on both the user side and the items side. We characterize it using the following four features: **S**: data *sparsity*, related to the original cold-start problem; **V**: *volatility*, or the degree of variation in the object of interest; **I**: object *identity*, due to different technical [177] or law regulation related problems complicating correct identification; **P**: '*personas*', or the different types of behavior expressed by one user in different situations.

The *User Continuous Cold Start Problem* (*UCoCoS*) can be characterized by:

- **S**: new or rare users;
- **V**: users' interests change over time;
- **I**: a failure to match data from the same user;
- **P**: users have different interests at different, possibly close-by points in time.

New users arrive frequently as shown in Figure 3.1(A), or may appear new when they do not log in or use a different device so we would fail to match their identity. Some websites are prone to extreme sparsity in user activity when items are purchased only rarely, such as travel, cars etc. Most users change their interests over time (volatility), e. g. movie preferences evolve, or travel needs change. On even shorter timescales, users have different personas. Depending on their mood or their social context, they might be interested in watching different movies. Depending on the weather or their travel purpose, they may want to book different types of trips as presented in Figure 3.1 (B).

Similarly we characterize *Item Continues Cold Start Problem* (*ICoCoS*):

- **S**: new or rare items;
- **V**: item properties or value change over time;

- **I**: a failure to match data from the same item;
- **P**: an item appeals to different types of users.

New items appear frequently in e-commerce catalogues, as shown in Figure 3.2 (A) for accommodations at Booking.com. Some items are interesting only to niche audiences, or sold only rarely, for example books or movies on specialized topics. Items can be volatile if their properties change over time, such as a phone that becomes outdated once a newer model is released, or a hotel that undergoes a renovation. Figure 3.2 (B) shows fluctuations of the review score of a hotel at Booking.com. Some items have different 'personas' in that they target several user groups, such as a hotel that caters to business as well as leisure travellers. When several sellers can add items to an e-commerce catalogue, or when several catalogues are combined, correctly matching items can be problematic so we run into an item identity problem.

### 3.3.2 Optimizing Destination List within CoCoS

To motivate our problem set-up, we introduce a Booking.com service which allows to find travel destinations based on users' preferred activities: the Destination Finder. Consider a user who knows what activities she wants to do during her holidays, and is looking for travel destinations matching these activities. This process is a complex exploratory recommendation task in which users start by entering activities in the search box as shown in Figure 3.3. The service returns a ranked list of recommended destinations [149].

The underlying data is based on 'endorsements' of users that have booked a hotel at some destination via the online travel agent in the past. After the users visited the destination, they are asked to endorse the place using a set of endorsements. Initially, the set of endorsements was extracted from users' free-text reviews using a topic-modeling technique such as LDA [34, 179]. Nowadays, the set of endorsements consists of 256 activities such as 'Beach,' 'Nightlife,' 'Shopping,' etc. These endorsements imply that a user liked a destination for particular characteristics. Two examples of the collected sets of endorsements for two destinations 'Bangkok' and 'London' are shown in Figure 3.4. As an example of the multi-criteria endorsement data, consider three endorsements: $e_1$ = *'Beach'*, $e_2$ = *'Shopping'*, and $e_3$ = *'Family Friendly'* and assume that a user $u_j$ after visiting a destination $d_k$ (e. g. 'London') provides the review $r_i(u_j, d_k)$ as:

$$r_i(u_j, d_k) = (0, 1, 0). \tag{3.4}$$

This means our user ranks London for the 'Shopping' activity only. However, we cannot conclude that London is not 'Family Friendly', i. e. negative user opinions are hidden. In contrast to the ratings data of the traditional recommender systems setup, we are dealing with multi-criteria ranking data. Destination Finder is a good example of the service which is working under the *CoCoS* settings from both sides: users and items.

***UCoCoS* at Destination Finder** It is used to plan holidays, so many users visit it infrequently because they have only one or two vacations per year, leading

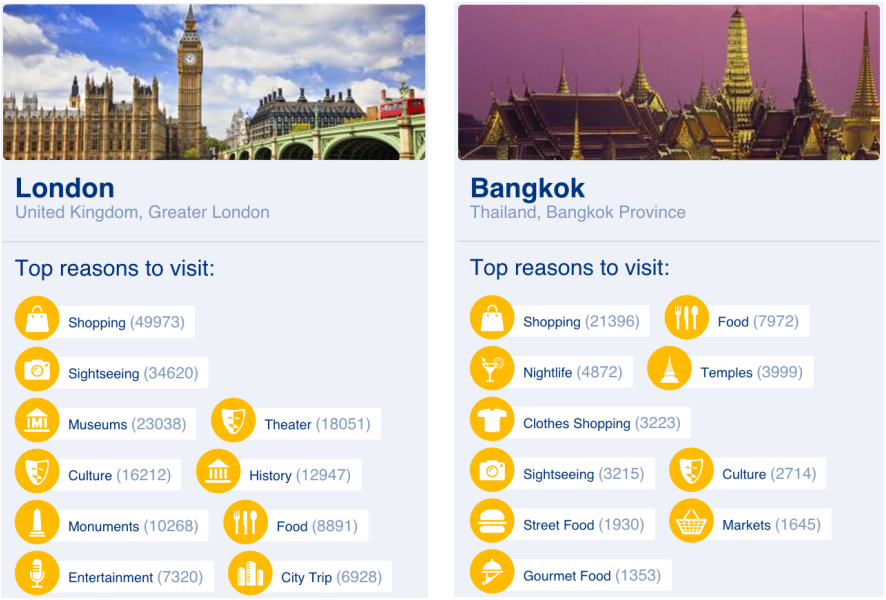



Figure 3.4: The Destination Finder endorsement pages of London and Bangkok.

to the sparsity problem. Since users interact with service rarely—many changes can happen and they might shift their preferences from backpacking activities to family friendly places. Users can use different devices to search over Destination Finder without login to the system, so user matching is an actual problem. Users can express different types of preference while planning trips, e. g. they might go to a family friendly resort while traveling with children and look for 'Shark Diving' while planning holidays alone, so we need to deal with different user 'personas'.

***ICoCoS* at Destination Finder** The list of destinations is growing continuously over time because users share their experience about new places, so we run into the item sparsity problem. User reviews for destination depends on contextual information. For example, the resort 'The Hague at North Sea' is widely endorsed for the activity 'Beach' during summer, but not during winter, so we run into the item volatility. Moreover, destination might change over time, e. g. a new aquarium is build and users start to endorse a place for it. Some destinations have different 'personas' in which they target several user groups, such as a destination which can be family friendly but at the same time has rich night live. Therefore, we have places that are expressing different 'personas'.

These aspects of *CoCoS* at Destination Finder can be addressed partially by taking context into account. We propose that the described multi-criteria endorsements can be enhanced by contextual information. We build a contextual ranker for recommending destinations, whereas the current live systems uses an

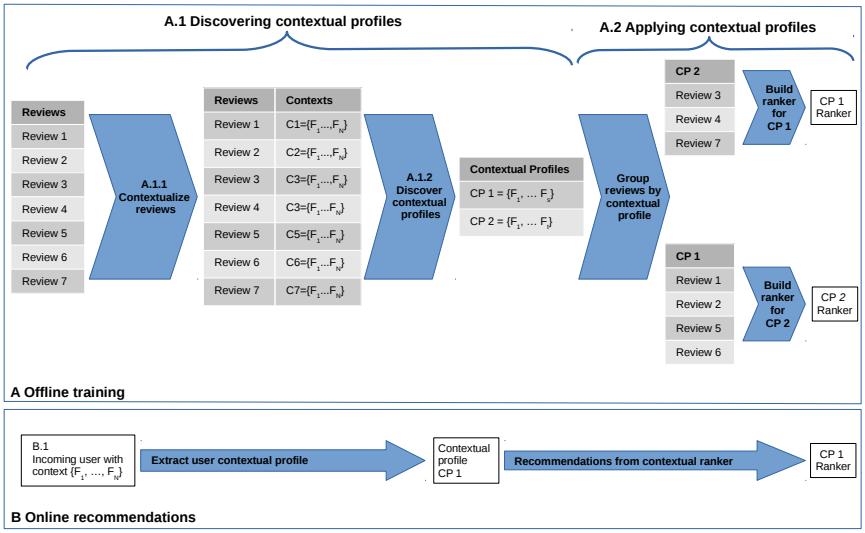



Figure 3.5: An overall framework for discovering multidimensional contextual user profiles.

advanced non-contextualized ranker.

To summarize, we introduced the continuous cold start problem, and characterized the user and item sides of the *CoCoS*. We also introduced the Destination Finder setup that we used in this work as follows:

- we have a set of geographical destinations such as 'Paris', 'London', 'Amsterdam' etc.;
- each destination is ranked by users who visited the place using a set of endorsements under some situation (which can be described by a set of contexts).

In the setting of *CoCoS*, our main goal is to find ways to map any incoming user, without assuming prior history or explicit profiles, to some cluster of like-minded previous users using only contextual data. In the next section, we will discuss how to discover such contextual user profiles.

## 3.4 Multidimensional Contextual User Profiles

In this section we will study our ***RQ 3.2:*** *How to define and discover contextual user profiles from multi-criteria ranking data in an unsupervised setup?* We present an overview of our framework for discovering multidimensional contextual user profiles (CUPs), as outlined in Figure 3.5. It has two main stages: *offline* (**A**), and *online* (**B**). The discovery of multidimensional CUPs (**A.1**) happens during the offline stage and is described in Section 3.4.1. The process of using

discovered CUPs is as follows: (**A.2**) during the *offline stage*, we apply the set of discovered CUPs to learn a customized ranker; and (**B**) during the *online stage*, we assign incoming users to one of the CUPs. The process of using CUPs is presented in Section 3.4.3. Section 3.4 defines CUPs in a generic way. In Section 3.5 we show how the framework can be applied to the Destination Finder.

### 3.4.1 Defining Contextual User Profiles

Apart from the reviews, as defined in Equation 3.4, there is additional contextual information about the *situation* in which users made their choice (to consider or not to consider the suggested destination), e. g. the geographical location, the time (when a user is using Destination Finder), the users' device type, or the referral (where is a user coming from). We adopt the definition of the context as described in Equation 3.2.

In many real world RS it is not feasible to track user identity information $u_j$ for several reasons: **(1)** privacy issues: only a limited part of the user interaction history can be stored; **(2)** the cold-start problem: when a new user comes without prior history of interaction with the system; **(3)** a user does not have to be logged in: so we cannot make use of his interaction history. However, we would like to predict user preferences in order to supply him with suitable recommendations. Therefore, we want to detect a list of typical user situations using contextual information. This type of situational information we call CUPs.

Contextual information can be represented as a $n$-dimensional space where the dimensions are the set of contextual features, $\{F_n\}_{n=1}^{N}$, and the coordinates for each dimension are the contextual categories, $\{v_m\}_{m=1}^{M}$. For example, the contextual feature $F_1$, 'User Device', is represented by the following contextual categories:

$$F_1 = \{v_1 = \text{'Mobile'}, v_2 = \text{'Tablet'}, v_3 = \text{'PC'}\}. \tag{3.5}$$

To simplify the notation we rewrite Equation 3.5 as:

$$F_1 = \{F_{11} = \text{'Mobile'}, F_{12} = \text{'Tablet'}, F_{13} = \text{'PC'}\}. \tag{3.6}$$

The 3-dimensional example (cube) of contextual space is presented in Figure 3.6 (A) where we have three dimensions: $\{F_1 = \text{'OS'}, F_2 = \text{'Browser'}, F_3 = \text{'Time'}\}$.

A *contextual user profile* is a *region* in the $n$-dimensional contextual space that represents 'typical' user behavior. When a user visits our service we can map him to one of the CUPs and use this insight into his preferences to improve the quality of the service, i. e. serving better travel recommendations in the Destination Finder.

### 3.4.2 Discovering Contextual User Profiles

We now discuss in more detail the process of discovering CUPs, as outlined in Figure 3.5 (A.1). The review entities as defined in Equation 3.4 can be contextualized, i. e., extended by multidimensional contextual information $C$ as depicted

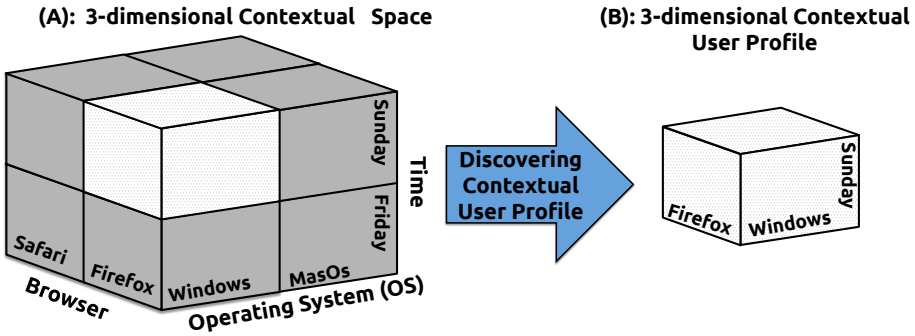



Figure 3.6: An example for discovering a contextual user profile from 3-dimensional contextual space. The $3D$ contextual space can be visualized as a cube (A), of which the contextual user profile is a cube region (B).

in Figure 3.5 (A.1.1). We use the context definition presented in Equation 3.6. The contextual review $r_i$ has the following form:

$$r_i(u_j, d_k) = (e_1, \ldots, e_X, F_{11}, \ldots, F_{NM}), \tag{3.7}$$

where:

1. $u_j$ is user information that is not stored explicitly, but in our setup we have contextual information regarding how a review is made;
2. $d_k$ is a destination which a user $u_j$ ranks using multi-criteria endorsements;
3. $e_1, \ldots, e_X$ are endorsements represented as binary values;
4. $F_{11}, \ldots, F_{NM}$ are contextual features represented in a binary way. For example, if a user is using a device with 'Windows' as OS and a 'Firefox' browser on Sunday, then the context vector is $(1, 0, 0, 1, 0, 1)$.

In our setup we combine CARS and MCRS presented in Equation 3.1 and 3.3 accordingly. A key difference to standard settings is that we are dealing with sparse multi-criteria ranking data, not with ratings. Therefore, negative user opinions are hidden from us.

Our assumption is that users give similar endorsements in similar situations, and that we can represent it by a subspace of contexts. In order to enrich the contextual space, we use the review entities with endorsements as an additional dimension to the $n$-dimensional contextual space. We apply clustering techniques to discover *'closely connected'* regions in the contextual space. After finding the contextual regions in the extended $(n + 1)$-dimensional cube we eliminate the endorsement dimension in order to derive CUPs which consist solely of contexts. This allows us to map new incoming users to CUPs.

The CUP is represented as an agglomeration of a discovered region. For example, if a clustering technique is applied then a cluster center would be an

example of CUP, as we will explain in Section 3.5.2. In the example in Figure 3.5 (A.1.2), we discover two CUPs: $CP_p$ and $CP_q$. The choice of the clustering method depends on the type of application. We detail the application to the Destination Finder in Section 3.5.2. Next, we discuss how the discovered CUPs can be used for ranking suggested destinations.

### 3.4.3 Using Contextual User Profiles

The process of using discovered CUPs can be divided into two main parts, see Figure 3.5: **(A.2)** *offline* application of CUPs; and **(B)** *online* mapping of an incoming user to one of the CUPs.

During the offline stage, the set of CUPs can be used for splitting reviews in order to build a set of contextual rankers $\{R_l\}_{l=1}^{L}$ where $L$ is the number of discovered CUPs. Our assumption is that a set of contextual rankers serves 'better' (more suitable) results than a base ranker $R_b$ which is trained based on all reviews.

During the online stage, an incoming user is mapped to one of the CUPs. A user is represented by a vector of contexts as shown in Figure 3.5 (B.1). In order to map a user to one of the CUPs, $CS_1$ or $CS_2$, we can employ any distance metric $D$. The user would be assigned to the *'closest'* CUP, which is $CP_1$ in our example in Figure 3.5 (B). Then the user is supplied with a contextual ranker $R_1$ which corresponds to $CS_1$.

To summarize, we presented a general framework for discovering and using contextual user profiles. In principle, any contextual features can be used, including relatively shallow implicit situational context available in any online context. Also any ratings, reviews or other multi-criteria ranking data can be used, including travel endorsements. In the next section, we apply the framework to the Destination Finder application described in Section 3.3.

## 3.5 Contextual Travel Recommendations

In this section we will study our ***RQ 3.3:*** *How to apply contextual user profiles for the ranking of travel destinations in a continuous cold start setting?* We present an example how our framework for discovering contextual user profiles (CUPs) from Section 3.4 can be applied to the Destination Finder. First, we describe the data used for our experimental pipeline in Section 3.5.1. Second, we use a clustering technique to discover contextual user profiles (CUPs) in Section 3.5.2. Third, we present in Section 3.5.3: **(1)** how these CUPs can be used within a ranking technique based on Naive Bayes; **(2)** how the customized rankings are deployed for online user traffic. We use standard clustering and ranking methods, such as k-means and Naive Bayes, which scale well to the volume of data available. These methods are sufficient to answer our main question about the value of context-aware recommendations. Further optimization is left for future work.

### 3.5.1 Data

In the offline training stage, we use reviews collected within the year 2014. The final set contains in total 5,138,494 reviews. We derive two types of data from web logs as contextual information:

- **user agent data** which is presented by *four dimensions* such as *'Device Type'* with 5 contextual categories (*mobile, tablet* etc.), *'OS'* with 27 contextual categories (*Windows 8.1, Android, Linux, OS X* etc.), *'Browser'* with 114 contextual categories (*Internet Explorer 6, Firefox 30, Firefox 34, Safari 7* etc.), and *'Traffic Type'* with 16 contextual categories (*web, mobile browser, application* etc.);
- **time data** which is *one dimensional*: the day of the week (*Monday, Tuesday* etc.).

This type of contextual information is available in all typical web logs, and can be used to contextualize the reviews as presented in Figure 3.5 (A.1.1). In total, the contextual space has 5 dimensions with 397 coordinates. In the online testing stage, we run our experiment on live user traffic for 26,868 users.

### 3.5.2 Clustering Contextualized Reviews

We use a clustering technique to discover CUPs as shown in Figure 3.5 (A.1.2). We apply k-means clustering [116] over the set of contextualized reviews as presented in Equation 3.7. The number of clusters is selected based on Silhouette validation [194], which results in 20 clusters as the optimal number.

After obtaining the final set of clusters, we eliminate the endorsement dimension by projecting on the contextual space. We analyze the set of contexts that is associated with the clusters in order to derive the set of CUPs. Because of the projection on the contextual space, clusters may overlap in some contextual categories.

The cluster centers represent the set of discovered CUPs. We calculate weights for the coordinates of the cluster centres as the ratio of the *(number of times the coordinate $F_{nm}$ appears within cluster $C_i$)* divided by the *(number of times the coordinate $F_{nm}$ appears within all clusters)*. This weight $w_{ij}$ (where $i$ is a cluster identifier and $j$ is an identifier of a coordinate $F_{nm}$) shows how strongly the contextual category $F_{nm}$ is associated with cluster $i$: The closer $w_{ij}$ to 1, the stronger the association.

We employ a pruning technique over the obtained list of CUPs in order to clean up some obvious noise. If $w_{ij}$ is too small for some contextual category $F_{nm}$, then this category is distributed widely over all CUPs and it does not enhance our definition of CUP. After trails of experiments we empirically determine a threshold:If $F_{nm}$ has $w_{ij} < 0.2$, we do not include it into the CUPs. For example, sometimes contextual categories such as 'Monday', 'Tuesday' are removed because apparently they do not reflect any 'specific' behavior. By applying this pruning technique we ended up with 17 clusters. We present an example of two pruned CUPs in Table 3.1, which correspond to intuitions about similar users

Table 3.1: An example of two obtained cluster centers from real data. Cluster $i$ can be characterized as 'users coming from mobile devices' and Cluster $i+1$ as 'users coming from windows-based devices on Fridays and Sundays'.

| | Cluster | | |
|---|---|---|---|
| i-1 | i | i+1 | i+2 |
| ... | iPhone.OS.7.Chrome | Windows.Phone | ... |
| | iPhone.OS.5.Chrome | Windows.Vista | |
| | iPhone.OS.6.Chrome | Friday | |
| | Android.2.2 | Sunday | |
| | Android.2.2.Tablet | | |
| | Android.3.1.Tablet | | |
| | Android.4.0.Tablet | | |
| | Android.4.4.Tablet | | |
| | Android.2.1.Tablet | | |
| | Android.3.0.Tablet | | |
| | Android.4.1 | | |
| | Android.4.3.Tablet | | |

Table 3.2: Results of the Destination Finder A/B testing based on the number of unique users, searches and clicks. The contextual ranker does not significantly change conversion (probability to click at least once), but significantly increases clicks-per-user and click-though-rate (CTR). Significance is assessed as non-overlapping 95% confidence intervals.

| Ranker | Users | Searches | Clicks | Conversion | Clicks/user | CTR |
|---|---|---|---|---|---|---|
| Baseline | 13,306 | 34,463 | 6,373 | 21.7±0.7% | 0.479±0.012 | 18.5±0.4% |
| Contextual | 13,562 | 35,505 | 7,866 | 21.3±0.7% | 0.580±0.013 | 22.2±0.4% |

based on context. It may not be a priori clear why such a cluster provides meaningful context, but the clustering informs us that they have distinct interests and preferences.

Next, we will describe how the discovered CUPs can be applied to destination ranking.

### 3.5.3 Using Contextual User Profiles for Destination Ranking

As a primary ranking technique we use a Naive Bayes approach. We will describe its application with an example. Let us consider a user running the searching for 'Beach'. We need to return a ranked list of destinations. For instance, the ranking score for the destination 'Miami' is calculated using the following formula:

$$P(\text{Miami}, \text{Beach}) = P(\text{Miami}) \times P(\text{Beach}|\text{Miami}); \tag{3.8}$$

where $P(\text{Beach}|\text{Miami})$ is the probability that the destination Miami gets the endorsement 'Beach'. $P(\text{Miami})$ is a prior knowledge about Miami. In the simplistic case the prior would be a ratio of the number of endorsements for Miami to the total number of endorsements in our database.

If a user uses a second endorsement (e. g. + 'Food') the ranking score is calculated in the following way:

$$\begin{aligned} P(\text{Miami}, \text{Beach}, \text{Food}) = P(\text{Miami}) \times P(\text{Beach}|\text{Miami}) \\ \times P(\text{Food}|\text{Miami}); \end{aligned} \tag{3.9}$$

If our user provides $n$ endorsements, Equation 3.9 becomes a standard Naive Bayes formula.

We split our set of reviews according to the obtained clusters. Then we train a set of contextual rankers using the same approach as described in Equation 3.9 to obtain the customized rankers $R(C_i)_{i=1}^{17}$. This process can be mapped to the general framework presented in Figure 3.5 (A.2).

During the online stage, which is shown at general framework work-flow in Figure 3.5 (B), an incoming user to the Destination Finder is mapped to the closest CUP. As we use only situational context that does not change per session, we only have to assign our user to the nearest cluster once, and there is no need to update the assignment during the session. Then we use a ranker $R(C_i)$ which corresponds to CUP.

As a distance metric we use Euclidean distance, which deals well with the different nature of some of the clusters (e. g., some clusters capture aspects of the day of the week, and others capture aspects of the used devices). More advanced mapping of users as mixtures of CUPs is left to future work, as our main goal in this chapter is to determine the impact of contextual ranking.

To summarize, we described the use of the framework for discovering contextual user profiles for the Destination Finder. We contextualized reviews with user agent and time data. Our main goal is to determine the impact of contextual ranking, hence we use standard clustering and ranking methods. Specifically, we use k-means for clustering and Naive Bayes for ranking and we map incoming users to the nearest cluster based on euclidean distance. In the next section, we will present our experimental pipeline which involves online A/B testing at the major travel agent Booking.com.

## 3.6 Experiments and Results

In this section, we will study our ***RQ 3.4:*** *How effective are contextual profiles for real-world users of the destination finder system in terms of user engagement measures?* To test the effectiveness of contextualization, we perform experiments on users of Booking.com where an instance of the Destination Finder is running.

### 3.6.1 Research Methodology

We take advantage of a production A/B testing environment at the major online travel agency Booking.com. A/B testing randomly splits users to see either the baseline or the new variant version of the website, which allows to measure the impact of the new version directly on real users [155, 223]. As baseline we use a non-contextualized ranker corresponding to the live system. This is an optimized system, trained on a massive volume of traffic, and far superior to standard baselines such as popularity [149].

As our primary *evaluation metric* in the A/B test, we use clicks-per-user and click-through-rate (CTR) [161]. As explained in the motivation, we are dealing with an exploratory task and therefore aim to increase customer engagement. More clicks-per-user and higher CTR are signals that users click more on the suggested destinations and interact more with the system.

### 3.6.2 Results

Table 3.2 shows the results of our A/B test. We see that the contextual ranker does not significantly change conversion compared to the baseline non-contextual ranker, i. e. the probability for a user to click at least once remains the same. Thus, our recommendations do not influence the basic user intent of using the Destination Finder. In contrast, the contextual ranker significantly increases further user engagement after the first click: The CTR increases by absolute 3.7%, and both CTR and clicks-per-user increase dramatically by relative 20% and 23%, respectively. Our contextual recommendations invite users to perform more searches and click on more recommendations, both per search and per user. In total, users are significantly more engaged with the Destination Finder when presented with contextual recommendations.

We achieved this substantial increase in clicks with a simple contextualization using straightforward k-means clustering of reviews and a Naive Bayes ranker. Most computations can be done offline, and only simple calculations have to be performed online. Thus, our model could be trained on large data within reasonable time, and did not negatively impact wallclock and CPU time for the Destination Finder web pages in the online A/B test. This is crucial for a webscale production environment [155].

To summarize, we compared our contextual travel recommendations against the same non-contextualized ranker. This allowed us to compare the effect of contextualization independently of the underlying ranking. This is a hard baseline corresponding to the current live system applied to the exact same data. We observe a dramatic increase in user engagement, with click-through rates and clicks by users increasing by 20%. The simplicity of our contextual models enables us to achieve this engagement without significantly increasing online CPU and memory usage. The experiments clearly demonstrate the value of contextual profiles in a real world application.

## 3.7 Conclusions

This chapter investigated the common case in e-commerce websites relying on search and recommendation to satisfy their user's needs, yet standard personalization and recommender systems rely on rich user profiles but the majority of users are new or visit highly infrequently—we face a continuous cold start recommendation problem. We specifically studied this problem in the context of Booking.com, one of the largest travel websites, and its Destination Finder service.

Our first research question was ***RQ 3.1:*** *How to characterize the continuous cold start problem in travel recommendation?* We introduced and characterized the *Continues Cold Start Problem* (*CoCoS*) that happens when users (*UCoCoS*) or/and items (*ICoCoS*) remain 'cold' for a long time, and can even 'cool down' again after some time due to some external signals.

Our second research question was ***RQ 3.2:*** *How to define and discover contextual user profiles from multi-criteria ranking data in an unsupervised setup?* We presented a general framework for discovering and using contextual user profiles. Since we work in settings of *CoCoS* clients visit infrequently and have volatile interests, we cannot rely on historical user interactions. Mining situational profiles to which we can map an incoming user is an effective way to deal with data sparsity and changing user interests. In principle, any contextual features can be used, including relatively shallow implicit situational context available in any online context. Also any ratings, reviews or other multi-criteria ranking data can be used, including travel endorsements. Similar endorsement data is being used in a venue recommendation benchmark [227].

Our third research question was ***RQ 3.3:*** *How to apply contextual user profiles for the ranking of travel destinations in a continuous cold start setting?* We used the general framework for discovering contextual user profiles for the Destination Finder. As explicit situational context is not available in typical real world application, implicit cues from transaction logs used at scale can capture the essential features of situational context. We contextualized reviews with user agent and time data. Our main goal is to determine the impact of contextual ranking, hence we used standard methods, specifically k-means for clustering and Naive Bayes for ranking. We mapped incoming users to the nearest cluster based on Euclidean distance.

Our fourth research question was ***RQ 3.4:*** *How effective are contextual profiles for real-world users of the destination finder system in terms of user engagement measures?* We compared our contextual travel recommendations to a non-contextual ranker. This is a hard baseline corresponding to the current live system. Contextual user profiles can be created offline, resulting in a set of smaller models compared to the single, huge, non-contextual model, making contextual ranking available with negligible CPU or memory footprint. We observed an increase in user engagement, with higher click-through rates (20%) and higher clicks per user (21%).

Our general conclusion is that our contextual ranking approach shows a dramatic increase in user engagement over a non-contextual baseline, clearly demon-

strating the value of contextualized profiles in a real world application that suffers from *CoCoS*. We focused on an e-commerce setting, applicable to millions of online companies, where the continuous cold start is the rule rather than the exception. But also in settings such as the web search engines where interactions are frequent and rich profiles are typically available, our approach has large potential value. The problem of fast changing content is well-known [66]. Perhaps the fraction of new users is small, yet they may be important enough to warrant extra effort, think of new users considering a search engine switch [239].

Our future work is to further investigate the following directions. First, we plan to extend the contextual space, for example using the geographical location of the user. However, this is not straightforward since simple splitting using some ontological knowledge, e. g. country, can lead to very skewed distributions of traffic within the contextual features and fails to capture deeper relations in the data. More generally, we plan to look into unsupervised techniques for the context discovery, over a wider range of contextual conditions including aspects of the session at hand. Second, it is promising to extend our method of mapping incoming users to one of the discovered CUPs to a 'fuzzy' mapping in which a user can be assigned to two or more CUPs. This will allow to serve a personalized ranking based on the resulting mixture weights in the model, while still maintaining online efficiency. Third, we will look into possibilities of more efficient and accurate CUPs discovery techniques, looking also in adaptive models that take into account long term trends such as seasonal differences.

# Part II

# Predicting User Satisfaction with Intelligent Assistants

# Part II
# Predicting User Satisfaction with Intelligent Assistants

In the second part of this dissertation, we study methods to predict user satisfaction with the voice-controlled intelligent personal assistant Microsoft Cortana. intelligent assistants introduce a significant change in information access, not only by introducing voice control and touch gestures but also by enabling dialogues where the context is preserved. Specifically, Part II continues our work on the following research question:

> **RQ 3**: *How to discover users' behavioral aspects as contextual information?*

Chapter 4 conducts a user study on user satisfaction across the different intelligent assistants scenarios: i) device control, e. g. call a contact, check the calendar, access an application, etc.; ii) mobile web search; and iii) performing a complex task through a dialogue style interaction. Chapter 4 is based on [153].

Chapter 5 proposes an automatic method to predict user satisfaction with intelligent assistants that exploits all the interaction signals, including voice commands and physical touch gestures on the device, using ground truth data on dialogue style tasks obtained from the user study in the previous chapter. Chapter 5 is based on [152].

Finally, Chapter 6 investigates good abandonment, cases where a user may not click on any results on the SERP but still is satisfied, using mobile web search data from the user study in Chapter 4 and real snapshots of web traffic, using gesture interactions such as reading times and touch actions as signals for differentiating between good and bad abandonment. Chapter 6 is based on [244, 245].

# 4
# Intelligent Assistants

Voice-controlled intelligent personal assistants, such as Cortana, Google Now, Siri and Alexa, are increasingly becoming a part of users' daily lives, especially on mobile devices. They introduce a significant change in information access, not only by introducing voice control and touch gestures but also by enabling dialogues where the context is preserved. This raises the need for evaluation of their effectiveness in assisting users with their tasks. However, in order to understand which types of user interactions reflect different degrees of user satisfaction we need explicit judgments. In this chapter, we describe a user study that was designed to measure user satisfaction over a range of typical scenarios of use: controlling a device, web search, and structured search dialogue. Using this data, we study how user satisfaction varied with different usage scenarios and what signals can be used for modeling satisfaction in the different scenarios. We find that the notion of satisfaction varies across different scenarios, and show that, in some scenarios (e. g. making a phone call), task completion is very important while for others (e. g. planning a night out), the amount of effort spent is key. We also study how the nature and complexity of the task at hand affects user satisfaction, and find that preserving the conversation context is essential and that overall task -level satisfaction cannot be reduced to query-level satisfaction alone. Finally, we shed light on the relative effectiveness and usefulness of voice-controlled intelligent agents, explaining their increasing popularity and uptake relative to the traditional query-response interaction.

## 4.1 Introduction

Spoken dialogue systems [176] have been around for a while. However, it has only been in recent years that voice controlled intelligent assistants, such as Microsoft's Cortana, Google Now, Apple's Siri, Amazon's Alexa, Facebook's M, etc., have become a daily used feature on mobile devices.

Intelligent assistants enable new mechanisms of information access, that are very different from traditional web search. Figure 4.1 shows two examples of di-

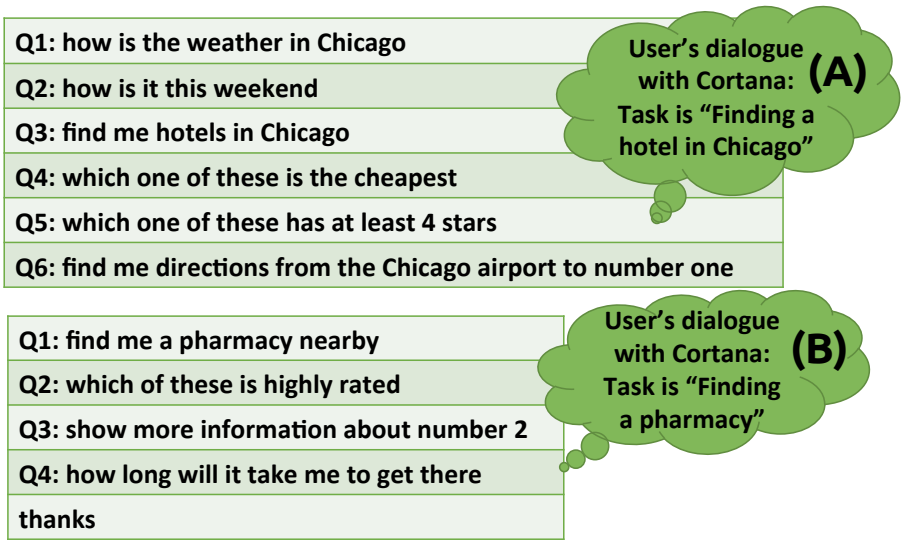



Figure 4.1: Two real examples of users' dialogues with an intelligent assistant: In the dialogue (A), a user performs a 'complex' task of planning his weekend in Chicago. In the dialogue (B), a user searches for the closest pharmacy.

alogues with intelligent assistants sampled from the interaction logs. They are related to two tasks: **(A):** searching things to do on a weekend in Chicago, and **(B):** searching for the closest pharmacy. Users express their information needs in spoken form to an intelligent assistant. The user behavior is different compared with standard web search because in this scenario an intelligent assistant is expected to maintain the context throughout the conversation. For instance, our user anticipates intelligent assistants to understand that their interaction is about 'Chicago' in the transitions: $Q_1 \rightarrow Q_2$, $Q_3 \rightarrow Q_4$ in Figure 4.1(A). These structured search dialogues are more complicated than standard web search, resembling complex, context-rich, task-based search [234]. Users expect their intelligent assistants to understand their intent and to keep the context of the dialogue —some users even *thank* their intelligent assistant for its service, as in example in Figure 4.1(B).

Users communicate with intelligent assistants through voice commands for different scenarios of use, ranging from controlling their device—for example to make a phone call, or to manage their calendar—to complex dialogues as shown in Figure 4.1. These interactions between users and intelligent assistants are more complicated than web search because they involve:

- automatic speech recognition (ASR): users communicate mostly through voice commands and it has been shown that errors in speech recognition negatively influence user satisfaction [121];
- understanding user intent: an intelligent assistant needs to understand user

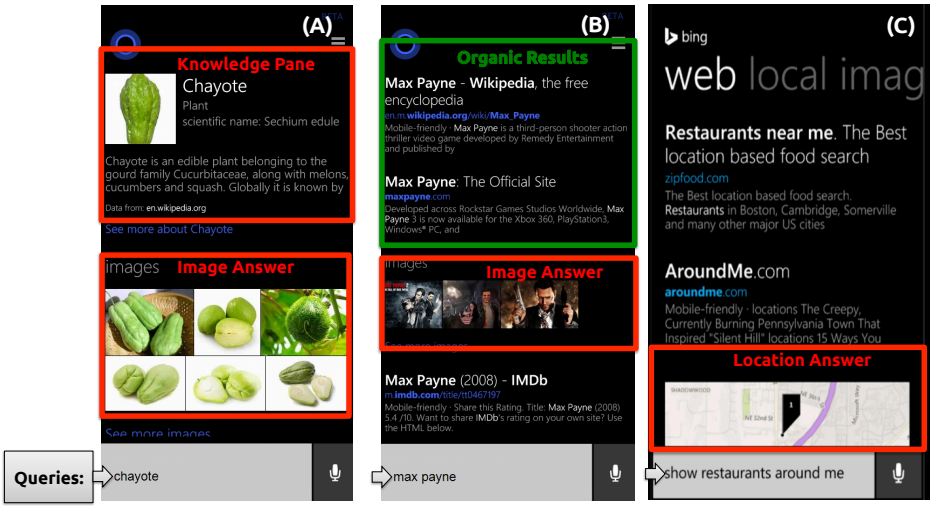



Figure 4.2: An example of mobile SERPs that might lead to 'good abandonment'.

intent in order to take action on the intended task, or to provide an exact answer when possible;

- dialogue-based interaction: users expect an intelligent assistant to maintain the context of the dialogue;
- complex information needs: users express more sophisticated information needs while interacting with intelligent assistants.

This prompts the need to better understand success and failure of intelligent assistant usage. When are users (dis)satisfied? How can we evaluate intelligent assistants in ways that reflect perceived user satisfaction well? Can we resort to traditional methods of offline and online evaluation or do we need to take other factors into consideration?

Evaluation is a central component of many web search applications because it helps to understand which direction to take in order to improve a system. The common practice is to create a 'gold' standard (set of 'correct' answers) judged by editorial judges [119]. In case of intelligent assistants, there may be no general 'correct' answer since the answers are highly personalized and contextualized (e. g., by the user's location, prior queries or interactions) to fit user information needs. Another way to evaluate web search performance is through implicit relevance feedback such as clicks and dwell time [11, 77, 97, 124, 125]. However, we know that user satisfaction for mobile web search is already very different [159].

In the examples in Figure 4.2, different types of answers are shown for queries such as *'Location Answer'*, *'Image Answer'* or *'Knowledge Pane Answer'*. Users can find required information directly on the search result page (SERP) and they do not need to perform any further interactions (e. g. clicks). So we cannot assume


Figure 4.3: An example of a 'simple' task with a structured search dialogue.

that users who do not interact with the SERP are dissatisfied. This problem of 'good' abandonment received a lot of interest in recent years [53, 54, 65, 166]. An example of a users' dialogue about *'weather'* is shown in Figure 4.3. All information about the weather is already shown to the users and they do not need to click. In case of structured dialogue search, the lack of standard implicit feedback signals emerges even more because users talk to their phones instead of making clicks. One example of this is the transition $Q_2 \rightarrow Q_3$ in Figure 4.1(B).

In light of the current work, this chapter aims to answer the following main research question:

*What determines user satisfaction with intelligent assistants?*

We break down our general research problem into five specific research questions. Our first research question is:

**RQ 4.1:** *What are characteristic types of scenarios of use?*

Based on analysis of the logs of a commercial intelligent assistant; and from previous work [123], we propose three types of scenarios of intelligent assistant use: **(1)** controlling the device; **(2)** searching the web; and **(3)** perform a complex (or 'mission') task in a dialogue interaction. We characterize key aspects of user satisfaction for each of these scenarios.

Our second research question is:

**RQ 4.2:** *How can we measure different aspects of user satisfaction?*

We set up user studies with realistic tasks derived from the log analysis, following the three scenarios of use, and measuring a wide range of aspects of user satisfaction relevant to each specific scenario.

Our third research question is:

**RQ 4.3:** *What are key factors determining user satisfaction for the different scenarios?*

In order to understand what the key components of user satisfaction are, we analyze output of our user studies for different intelligent assistants scenarios. We aim at understanding what factors influence user satisfaction the most: speech recognition quality, complexity of the task, or the amount of effort required to complete the task.

Our fourth research question is:

**RQ 4.4:** *How to characterize 'abandonment' in the web search scenario?*

*'Good abandonment'* makes it difficult to measure user satisfaction with web search scenario using conventional implicit feedback behavioral signals. We analyze the way in which users interact with the intelligent assistant following a web search; we characterize user satisfaction in general, and over the number of issued queries, and types of answers found.

Our fifth research question is:

**RQ 4.5:** *How does query-level satisfaction relate to overall user satisfaction for the structured search dialogue scenario?*

The structured search dialogue scenario introduced a new mechanism for users to interact with intelligent assistants which has not received a lot of attention in the literature. We analyze the data for the search dialogue interactions, and investigate satisfaction over tasks with increasing complexity; we consider how sub-task level satisfaction relates to overall task satisfaction.

The remainder of this chapter is organized as follows. Section 4.2 describes earlier work and background. Then, Section 4.3 introduces scenarios of user interaction with intelligent assistants, discusses differences and similarities in user behavior. Section 4.4 describes different types of user studies developed to evaluate user satisfaction for intelligent assistants different scenarios. Finally, Section 4.5 reports our results and findings. We summarize our findings, discuss possible extensions of the current work in Section 4.6.

## 4.2 Background and Related Work

In this section, we will discuss related work relevant to the research described in this chapter, covering three broad strands of research. First, we discuss research

on spoken dialogue systems in Section 4.2.1. Then, we present an overview of the methods for evaluating user satisfaction in web search systems in Section 4.2.2. Finally, we focus on user studies for the evaluation of intelligent assistants in Section 4.2.3.

### 4.2.1 Spoken Dialogue Systems

The main difference between traditional web search and intelligent assistants is their conversational nature of interaction with users. In the considered scenarios of usage of intelligent assistants, the technology can refer to the users' previous requests in order to understand the context of a conversation. For instance, in the dialogue (A) in Figure 4.1, the user asks for $Q_2$ and assumes that the intelligent assistant will 'remember' that he is interested in Chicago. Therefore, the spoken dialogue systems [176] are closely related to intelligent assistants because the spoken dialogue systems understand and respond to the voice commands in a dialogue form. This area has been studied extensively over the past two decades [228–230]. Most of these studies focused on systems that have not been deployed in a large scale and hence did not have the necessary means to study how users interact with these systems in real-world scenarios. However, intelligent assistants are different from traditional spoken dialogue systems because they also support interactions and 'understand' user intent. Furthermore, intelligent assistants display an answer which users can interact with and they are not purely based on speech—users can type in responses as well. From these perspectives, intelligent assistants are similar to multi-modal conversational systems [101, 236].

### 4.2.2 Evaluating User Satisfaction

User behavioral signals have been extensively studied and used for the evaluation of web search systems [9, 10, 95–97, 122, 134, 237]. Historically, the key objective of information retrieval systems is to retrieve relevant information (typically documents) or references to documents containing required information [196, 199]. Given this query-document relevance score, many metrics have been defined: MAP, NDCG, DCG, MRR, P@n, TBG, etc. [119]. For such setup we have a collection of documents and queries that are annotated by human judges. It is a common setup used at TREC[1]. In this case we evaluate system performance at the *query-level* for the pair $\langle Q, SERP \rangle$. Building such data collections needed for this type of evaluation is both expensive and time consuming. There is a risk that such collections may be noisy, given that third-party annotators have limited knowledge of an individual user intent.

User satisfaction is widely adopted as a subjective measure of search experience. Kelly [131] proposes a definition: *'satisfaction can be understood as the fulfillment of a specified desire or goal'.* Furthermore, recently researchers studied different metrics reflective of user satisfaction such as effort [249] and it has been shown that user satisfaction at the query-level can change over time [145, 148] due to some external influences. These changes lead to the necessity of updating

[1]Text REtrieval Conference: `http://trec.nist.gov/`

the data collection. Unfortunately, *query-level* satisfaction metrics ignore the information about a user's 'journey' from a question to an answer which might take more than one query [120]. Al-Maskari et al. [14] claim that *query-level* satisfaction is not applicable for informational queries – users can run follow-up queries if they are unsatisfied with the returned results; reformulations can lead users to an answer; this scenario is called *task-level* user satisfaction [66, 97]. Previous research proposed different methods for identifying successful sessions: Hassan et al. [97] used a Markov model to predict success at the end of the task; Ageev et al. [9] exploited an expertise-dependent difference in search behavior by using a Conditional Random Fields model to predict a search success – authors used a game-like strategy for collecting annotated data by asking participants to find answers to non-trivial questions using web search. On the other hand, situations when users are frustrated have also been studied: Feild et al. [75] proposed a method for understanding user frustration. Hassan et al. [98] and Hassan Awadallah et al. [99] have found that high similarity of queries is an indicator of an unsuccessful task. All described methods focus on analyzing user behavior when users interact with traditional search systems.

### 4.2.3 User Studies of Intelligent Assistants

In recent years voice-controlled personal assistants have become available to the general public. There are few studies researching intelligent assistants, and there is only one earlier paper that organizes a user study [123]. Jiang et al. [123] focus on simulated tasks for device control, as well as chat and web search, and identify satisfactory and unsatisfactory sessions based on features used in predicting satisfaction on the web, as well as acoustic features of the spoken request. Our work extends this study focusing on a wider range of scenarios of intelligent assistant use, including complex dialogues, and analyzing crucial aspects determining user satisfaction under these different conditions.

More broadly, intelligent assistants are often used for longer sessions and tasks that involve sub-tasks and complex interactions, and task complexity has been studied in many user studies. Wildemuth et al. [242] reviewed over a hundred interactive information retrieval studies in terms of task complexity and difficulty, and found that the number of sub-tasks, the number of facets, and the indeterminably were the main dimensions of task complexity. The structured search tasks we use in our study score high on these dimensions. Recently, Kelly [132] linked perceived task complexity with effort, suggesting that user satisfaction may depend on the amount of effort required to complete a complex task. We also look specifically at the role of effort relative to task-level user satisfaction.

To summarize, the key distinctions of our work compared to previous efforts are: we studied how users interact with intelligent assistants; we studied how we can use these interactions to understand *'good abandonment'*; we explored three main scenarios of user interactions with intelligent assistants and a definition of user satisfaction for these scenarios.

## 4.3 User Interaction with Intelligent Assistants

This section reports our study findings pertaining to the **RQ 4.1:** *What are characteristic types of scenarios of use?* In order to answer our research question we used the Microsoft intelligent assistant — Cortana. Historically, the scenario of controlling devices through voice commands was implemented first. It is described in detail in Section 4.3.1. From a user-satisfaction perspective, the main difference of this scenario compared with an information seeking task is that the 'right answer' is clear; in order to satisfy a user, an intelligent assistant needs to interpret requests correctly and give access to the correct functionality. In contrast, for information seeking tasks [115, 246] users exhibit different behavior. Cortana responds to a general search scenario by returning a variant of the Bing Mobile SERP, which may include answers or tiles from the knowledge pane as well as organic search results (see Figure 4.2); we discuss this scenario in Section 4.3.2. Another mechanism by which users interact with information systems that some intelligent assistants support is the 'structured search dialogue' (Figure 4.1). In this case, intelligent assistants are able to maintain the context of a conversation as the system engages with the user in a dialogue; it is definitely more complex (for the system) but at the same time a more natural (for the user) form of 'communication' between users and information systems. This scenario is presented in Section 4.3.3.

### 4.3.1 Controlling a Device

The first scenario of using intelligent assistants that we study is the direct access of on-device functionality – e. g., call a contact, check the calendar, access an app, etc. This scenario is useful because, ordinarily, it takes several actions to complete on existing smartphones. For example, in order to make a phone call, the user needs to first access a contact list on the phone and then identify the desired person. The ordinary process is time consuming, especially when the user is not familiar with the device. Instead, one can directly talk to the intelligent assistant to solve the problem, e. g., 'call Sam'. As long as the intelligent assistant can correctly recognize the user's words and task context, this largely reduces the user's effort.

Our user study includes the following types of on-device tasks that are popular in Cortana's usage logs:

- Call a person;
- Send a text message;
- Check on-device calendar;
- Open an application;
- Turn on/off wi-fi;
- Play music.

We group these tasks into one category because they share the similarity that users try to access these on-device functions through the intelligent assistants. These functions are normally not provided by the intelligent assistants, but offered by the device hosting it. In these tasks, intelligent assistants serve as a quick and efficient interface for accessing on-device functionality.

### 4.3.2 Performing Mobile Web Search

Another popular usage scenario for intelligent assistants is the general web search scenario. For this scenario, input can be either speech or text and there is no need for the system to be state-aware since it does not provide a multi-turn experience. During web search on mobile devices, the intent can be ambiguous. Therefore, the search result page (SERP) is very diverse and may include different types of answers such as:

- *'Answer Box'.* A box such as the knowledge pane (Figure 4.2(A)) or directions to a location (Figure 4.2(C)). These answer boxes are present for specific query intents.

- *'Image'.* In this case, just seeing an image may have satisfied a user's information need (e. g. Figure 4.2(A,B)).

- *'Snippet'.* The user's information need is satisfied by a snippet of text appearing below an organic search result (e. g. Figure 4.2(B)).

These different elements on a SERP can all lead to user satisfaction. For instance, the knowledge pane might contain the answer that the user is looking for or a user may be satisfied by the text in a snippet.

In some cases, the SERP is able to directly satisfy the user's information need and it can lead to the absence of one of the most studied user interaction signals (i.e. clicks on the SERP). Previous work on general web search has shown that presenting these types of answers affects user behavior [159] and leads to 'good abandonment' [54, 166] where the user appears to have abandoned the results but was actually satisfied without the need to engage with the SERP using clicks.

### 4.3.3 Structured Search Dialogue

In the structured search dialogue scenarios, the users are engaged in a conversation with the system using voice as we show in Figure 4.1. Cortana returns a structured answer that is distinguishably different from the usual SERP (Figure 4.2). The key component of this scenario is the ability of the intelligent assistant to maintain the context of the conversation. Examples of tasks where this scenario is activated include places (e. g. restaurants, hotels, travel, etc.) and weather. There are two types of dialogues that fall under this scenario: single task search dialogues and multi-task search dialogues.


Figure 4.4: An example of a structured search dialogue (multi-task search dialogue).

### Single Task Search Dialogues

Single task search dialogue have one underlying atomic information need and mostly consist of one query and one answer. An example of multi-task search dialogue is the weather-related task shown in Figure 4.3. multi-task search dialogues can be very similar to web search scenarios. We expect that they can be evaluated using a paradigm of *query-level* satisfaction because such dialogue usually consists of one query and one answer.

### Multi-Task Search Dialogues

Multi-task search dialogue consist of multiple interactions with Cortana that lead towards one final goal (e. g. 'find a place for vacation'). The final task can be divided into sub-tasks; the complexity of 'missions' is dependent on the need to understand the context of the conversation.

The example of a places-related multi-task search dialogues is presented in Figure 4.4. A user makes the following transitions:

- **(1)** 'asking for a list of the nearest restaurant' → **(2)** 'sorting the derived list to find best restaurants';

  *(**Comment for the transition** 1 → 2 **:** Cortana 'knows' that a user is working on the same list of restaurants)*

- **(2)** → **(3)** 'selecting the restaurant from the list and asking for the directions';

  *(**Comment for the transition** 2 → 3 **:** Cortana 'knows' that a user is working with the sorted list of restaurants)*

This type of interaction can be viewed as a sequence of user requests ('user journey towards a information goal') where each request is a step towards user satisfaction or frustration. Much of the frustration happens when Cortana is not able to keep the context and users need to re-attempt the task from the start. Going back to the example in Figure 4.1 (B), if Cortana did not carry the context across the transition $Q_3 \rightarrow Q_4$ (e. g. due to automatic speech recognition (ASR) error) then the user has to restart the task. Overall, user satisfaction goes down dramatically in this case, especially because the mistake happens at the end of the session.

To summarize, in this section, we categorized three distinct scenarios of user interactions with intelligent assistants. Cortana was used as an intelligent assistant example. We discussed difficulties in evaluating user satisfaction in each of these scenarios. For the controlling a device scenario, users' requests cannot be characterized by information needs. In order to satisfy users' needs the system is required to recognize their speech correctly and map a request to the right functionality. The web search and structured search dialogue are more complex because a comprehensive information seeking process is involved. The effect of good abandonment makes it difficult to measure user satisfaction. The structured search dialogue is a novel way of users' interactions that support complex tasks which consist of more than one singular objective. We refer to these complex tasks as multi-task search dialogues.

## 4.4 Designing User Studies

This section addresses **RQ 4.2:** *How can we measure different aspects of user satisfaction?* by describing the design of user study to collect user interaction data and ratings for different intelligent assistant scenarios. We start by characterizing the participants of our study in Section 4.4.1 followed by a description of the environment of the studies in Section 4.4.2. The general procedure for the study is presented in Section 4.4.3. Then, we present the detailed tasks and user study procedure for the different scenarios separately: device control in Section 4.4.4, structured search dialogue in Section 4.4.6, and mobile web search in Section 4.4.5. While designing the user study tasks we follow two requirements: **(1)** the simulated tasks should be realistic and as close as possible to real-world tasks ; **(2)** following Borlund [38] we construct the simulated tasks so that participants could relate to them and they would provide '*enough imaginative context.*'

### 4.4.1 Participants

We recruited 60 participants through emails sent to a mailing list of an IT company located in the United States. All participants were college or graduate students interning at the company or full time employees. They are reimbursed $10 gift card for participating in an experiment. The average age of participants is 25.53 years ($\pm$ 5.42). The characteristics of the participants regarding gender (A), field of education (B) and native language (C) are presented in Table 4.1.

Table 4.1: Demographics of the user study participants: gender (A), native language (B), and field of education (C)

| **Gender** | | **Native language** | | **Field of education** | |
|---|---|---|---|---|---|
| *Male* | 75% | *English* | 55% | *Computer science* | 82% |
| *Female* | 25% | *Other* | 45% | *Electrical engineering* | 8% |
| | | | | *Mathematics* | 7% |
| | | | | *Other* | 3% |

### 4.4.2 Environment

Participants performed the tasks on a Windows phone with the latest version of Windows Phone 8.1 and Cortana installed. If the task needed to access some device resources, functions or applications (e. g. maps), they are installed to make sure users would not encounter problems. The experiment was conducted in a quiet room, so as to reduce the disturbance of environment noise. Although the real environment often involves noise and interruption, we eliminate those factors to simplify the experiment.

### 4.4.3 General Procedure

The participants were first asked to watch a video introducing the different usage scenarios of Cortana, and then complete a background questionnaire with demographics and previous experience with using intelligent assistants. Then, they work on one training task and eight formal tasks. We instructed participants that they could stop a task when they had accomplished the goal or if they became frustrated and wanted to give up. Finally, they were asked to answer an extensive questionnaire on their experience and share further details during a short interview.

For each task, we asked participants to listen to an audio recording that verbally described the task objective. We did not show the participants the task description while they were working on the task, because in an earlier pilot study, many participants directly used the sentences shown in task descriptions as requests. We strongly want to avoid such outcome because our goal is to simulate real user behavior. After completing the task, participants were directed to the questionnaires. The questions depend on the objectives of the experiment and vary per user study. Participants answered all questions using a standard 5-point Likert scale.

### 4.4.4 User Study for Controlling Device

The first user study is to conduct the most basic scenario–controlling a device. We will now describe the tasks and the specific procedure for this study.

#### Tasks

In total we develop nine device control tasks. We rotated the assignment of tasks using a Latin square such that 20 participants worked on each unique task. Some examples of these tasks are:

- *Ask Cortana to play a song by Michael Jackson (a song by the artist is downloaded on the device prior to the task).*
- *You are on your way to a meeting with James, but will be late due to heavy traffic. Send James Smith a text message using Cortana and explain your situation.*
- *Create a reminder for a meeting with James next Thursday at 3pm.*
- *Ask Cortana to turn off the Wi-Fi on your phone.*
- *Ask Cortana to open WhatsApp (the name of a popular App, and the App is installed on the device prior to the task).*

#### Procedure

The instructional video about the controlling device scenario is about 2 minutes long. Our informal observation is that the video instructions were effective and felt like a natural extension of the speech interaction of the study, framing the study for the participants better than written instruction would do. When the participants worked on this user study, they were asked to use mostly voice for interactions. After terminating a task, they answered questions regarding their experience, including:

1. *Were you able to complete the task?*
2. *How satisfied are you with your experience in this task?*
3. *How well did Cortana recognize what you said?*
4. *Did you put in a lot of effort to complete the task?*

The total experiment time was about 20 minutes.

### 4.4.5 User Study for Web Search

The next use-case for the user study is general web search. There has already been significant research involving search on mobile phones [159, 211]; however, *'good abandonment'* in mobile search has had limited investigation. It is a particularly interesting problem to investigate as queries in mobile search have been described as *quick answer types* and previous research has shown that users formulate mobile queries in such a way as to increase the likelihood of the query being satisfied directly on the SERP [166]. For this reason, in this user study we choose to focus on tasks that have an increasing likelihood of leading to good abandonment. We first introduce the used tasks and then the specific procedure for this study.

### Tasks

The tasks for web search were designed to encourage answer-seeking behavior and increase the likelihood of good abandonment. The tasks involved:

- *A conversion from the imperial system to the metric system.*
- *Determining if it was a good time to phone a friend in another part of the world.*
- *Finding the score of the user's favourite sports team.*
- *Finding the user's favourite celebrity's hair colour.*
- *Finding the CEO of a company that lost most of its value within the last 10 years.*

After data cleaning, we retained the data from 55 users who completed a total of 274 tasks, 194 of which were labeled as SAT, while the remaining 70 were labeled as DSAT. There were a total of 607 queries for these tasks of which 576 were abandoned, thereby indicating that we were successful in designing tasks that had a higher potential of leading to good abandonment.

### Procedure

The user study starts with the instructional video (about 3 minutes long) that contains an example task for general web search. After completing each task, users were asked:

1. *Were you able to complete the task?*
2. *Where did you find the answer?*

   (**Suggested Answers:** *In an answer box*; *On a website that I visited*; *In a search result snippet*; *In an image.*)
3. *Which query led you to finding the answer?*

   (**Suggested Answers:** First; Second; Third; Fourth or later)
4. *How satisfied are you with your experience in this task?*
5. *Did you put in a lot of effort to complete the task?*

The purpose of the second question was to allow us to better understand where users find information that they are looking for. The option *'On a Website that I visited'* means a user clicked on a search result and visited a website to find the information that they were looking for.

The purpose of the third question was to allow us to tie a success event within a task to a specific query for future evaluation. We did not ask users about ASR quality because we gave users the option of using text input instead of speech. The reason for doing this is that, since we wanted to study good abandonment, we

tried to reduce the level of frustration due to speech recognition errors. However, even though that was the case, we still found that most of the participants used voice input because they found it more convenient. The total experiment time was about 20 minutes.

### 4.4.6 User Study for Structured Search Dialogue

This Section introduces the design of the user study to explore user satisfaction for the structured search dialogue. Again, we first describe the way we create tasks for our user study and tasks examples and then we describe the specific procedure for this study.

#### Tasks

In order to come with the list of tasks for participants, Cortana's logs (over 400K requests) are analyzed. We look at the terms distribution to get an idea for what kind of places users are looking for. Based on our analysis we come up with eight tasks, designed to cover a large portion of topics used by Cortana's users.

Among these eight tasks we have:

- *(A) one single task search dialogue task that is related to the weather where almost all participants are satisfied;*
- *(B) four multi-task search dialogues that include two sub-tasks;*
- *(C) three multi-task search dialogues tasks that require at least three switches in a subject.*

Tasks are given to participants in a free/general form in order to get query diversity and stimulate use satisfaction or frustration with returned results. For instance, let us consider the multi-task search dialogues with 3 sub-tasks: *'You are planning a vacation. Pick a place. Check if the weather is good enough for the period you are planning the vacation. Find a hotel that suits you. Find the driving directions to this place.* By giving a free-form task we stimulate the information need of participants (they need to come up with their own goal and they are more involved in the tasks) so this scenario should lead to satisfaction or frustration. For instance, out of 60 responses for the described task we get 46 unique places.

As a result of free-form task-formulation we obtained a diverse query set, characterized by the following: participants performed a total of 540 tasks that incorporated $2,040$ queries, of which $1,969$ were unique and the average query-length is 7.07. The single task search dialogue generated 130 queries in total; five (B)-type tasks generated 685 queries; three (C)-type tasks generated $1,355$ queries.

#### Procedure

The introductional video for this user study is about 4 minutes long and informs participants how to use the structured search dialogue. During this user study,

we instruct participants to verbally interact with Cortana. We instruct them to use text input only if Cortana does not understand their requests more than three times. Only after completing a task, they are redirected to questions regarding their experience in this task session. For multi-task search dialogues, users are asked to indicate their satisfaction with both the sub-tasks and the whole task in general. In order to stimulate participant involvement in the tasks, we asked them to answer clarifying questions. For instance, if the task was *'what is the weather tomorrow'*, the user also needed to indicate the temperature; this way we keep participants engaged.

Participants answer the following four questions after completing the tasks:

1. *Were you able to complete the task?*

2. *How satisfied are you with your experience in this task in general?*

   If the task has sub-tasks participants indicate their graded satisfaction e. g. **a.** *How satisfied are you with your experience in finding a hotel?* **b.** *How satisfied are you with your experience in finding directions?*

3. *Did you put in a lot of effort to complete the task?*

4. *How well did Cortana recognize what you said?*

The total experiment time was about 30 minutes.

To summarize, we described how we designed the user study with the objective of understanding user satisfaction with different scenarios of intelligent assistants, measuring relevant variables as speech recognition quality, task completion, and the effort taken. The introductory videos designed for the user study are available.[2] Detailed descriptions of the tasks and the recording on the tasks can be accessed.[3]

## 4.5 Results and Findings

This section presents the results and findings from the user studies, investigating our three remaining research questions (**RQ 4.3**, **RQ 4.4** and **RQ 4.5**). In Section 4.5.1, we focus on the user satisfaction relative to the different usage scenarios, and in relation to other measures like the speech recognition, task completion and effort taken. In Section 4.5.2, we analyze 'good abandonment' in web search, in short sessions where answers may be shown without the need for further interaction. In Section 4.5.3, we focus on structured search dialogues and how session- or task-level satisfaction relates to subtask-level satisfaction for longer sessions.

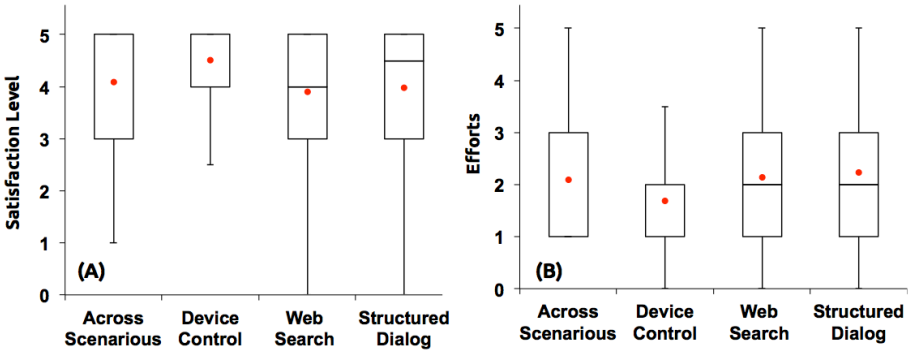



Figure 4.5: User satisfaction (A) and effort (B) across scenarios and in three discussed scenarios separately. Mean is red dot. Median is horizontal line.

### 4.5.1 Scenarios of Use

We will now investigate **RQ 4.3:** *What are key factors determining user satisfaction for the different scenarios?* The scenarios of use differ considerably in terms of complexity, session duration, type of outcome, and more, suggesting that different factors may play a role in determining user satisfaction.

We first discuss the distribution of user satisfaction across all aforementioned mechanisms of intelligent assistant use, both over the entire session and broken down by scenario—device control, web search on a mobile device, and structured search dialogue —which is presented in Figure 4.5(A). The user satisfaction is very high with means around 4 on a 5-point scale, both overall sessions and for each of the three scenarios. The high level of satisfaction showcases the maturity of the current generation of intelligent assistants, and explains the increasing adoption. As a case in point, many participants had (almost) never used the service, and were impressed by its effectiveness. We can see that user satisfaction with the device controlling tasks (mean of 4.5) is somewhat higher on average than with the information seeking tasks (mean of 3.7), plausibly because the information seeking tasks are open domain and more complex.

We also show the distribution of user effort, both across scenarios and separately, in Figure 4.5(B). Here we see relatively low scores for effort overall, consistent with high levels of satisfaction[4]. When we break down the effort over the scenarios, a similar picture emerges as with user satisfaction: participants spend more effort on search tasks, especially structured search.

We now perform a correlation analysis of user satisfaction and its components. Table 4.2 presents the correlation of user satisfaction with **(1)** speech recognition quality (ASR), **(2)** task completion (participants indicate if they are able to com-

[2] `https://goo.gl/6Gv5Y5`

[3] `https://goo.gl/0jXu2J`

[4] To be precise, this is based on the response to the question 'was a lot of effort required to complete the task?', measured on a Likert scale, where low scores indicate disagreement with the statement, hence that not much effort was required.

Table 4.2: Correlations of user satisfaction with other measures: ASR quality, Task Completeness, User Efforts. The sign * stands for statistically significant results ($p < 0.05$)

| Measures | All | Device Control | Web Search | Struct. Dialogue |
|---|---|---|---|---|
| SAT vs. ASR | 0.57* | 0.57 | –† | 0.56* |
| SAT vs. Completion | 0.18* | 0.59* | 0.10 | 0.10* |
| SAT vs. Effort | -0.75* | -0.64* | -0.65* | -0.80* |
| ASR vs. Completion | -0.22* | -0.27* | –† | -0.19* |
| ASR vs. Effort | -0.54* | -0.56* | –† | -0.51* |
| Completion vs. Effort | -0.11* | -0.39* | -0.08* | -0.05* |

†ASR was not calculated for web search as both spoken and typed queries were used.

plete the suggested task), and **(3)** effort spent (participants report the perceived effort to complete the task). We also look at the correlation between effort and completion. An obvious finding is that user satisfaction depends on ASR quality which is consistent with previous research [123]. Hence ASR quality is a key component of user satisfaction. We find a more interesting pattern for task completion: there is a high correlation with satisfaction for device control, but a low correlation for the information seeking scenarios. This suggests that users are able to find the required information and complete their tasks even in cases where their user satisfaction is suboptimal. And the strong negative correlation between satisfaction and effort shows that users spend a considerable amount of effort to complete their task.

This has important methodological consequences: we cannot equate 'success' in terms of task completion with user satisfaction for the informational scenarios, and have to incorporate the effort taken as a key component of user satisfaction across the different intelligent assistant scenarios. This finding is in line with recent work on task complexity or difficulty and effort, which postulates that satisfaction is low (high) for tasks that take more (less) effort than expected [132]. In addition, ASR quality is of obvious influence on user satisfaction. However, speech recognition is improving constantly and reached the levels that users can recover from misrecognition within a dialogue and still complete their task, at the cost of some extra effort and frustration.

### 4.5.2 Good Abandonment for Web Search

We continue with investigating our **RQ 4.4:** *How to characterize 'abandonment' in the web search scenario?* Whilst intelligent assistants can encourage highly interactive sessions, many results are provided as answers in speech or on the screen, requiring no further interaction of the user (e. g. no need to open a web page and read further to extract the requested information). Hence many sessions stop without an explicit user action, making it hard to discern good and bad search

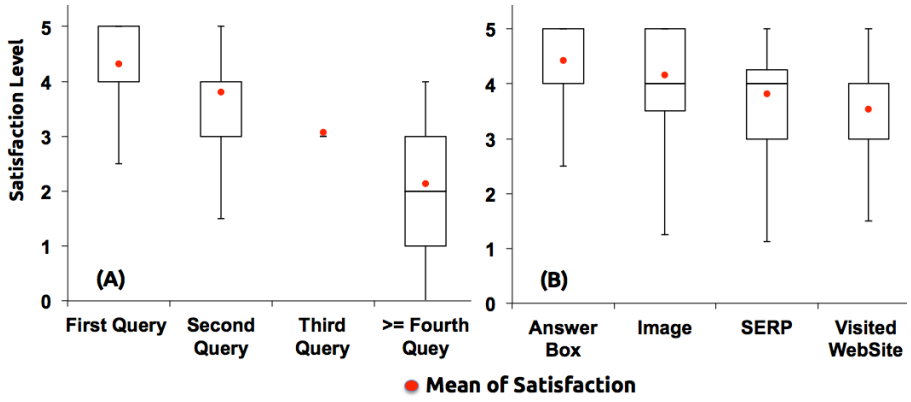



Figure 4.6: User satisfaction in the web search scenario: satisfaction over the number of queries that users run to find a required answer (A), and over where users find a required answer (B). The mean is represented by the dot and the median is the horizontal line.

abandonment from interaction log data.

We analyze the phenomenon of *'good abandonment'* from two perspectives: **(1)** the session length and **(2)** where users find the answer addressing their intent. Figure 4.6(A) presents the dependency of user satisfaction and how much effort was required to find an answer. Effort is associated with the number of queries that participates issued to find the required information. Our observations suggest that user satisfaction is higher if users use fewer queries to reach their goal. Figure 4.6(A) suggests that if users cannot find an answer after their first query their satisfaction goes down dramatically. Longer sessions lead to user frustration; however, task completion levels are high for the web search scenario, indicating that unnecessary effort was spent in completing the task.

Figure 4.6(B) shows the dependency of user satisfaction on the place where users find the desired answer. Furthermore, users are more satisfied if they can find a required result directly ('Answer Box' and 'Image') without the need to interact with the SERP such as **(1)** finding an answer in snippets ('SERP'); **(2)** clicking on SERP ('Visited Website'). Hence, cases without further interaction ('Answer Box' and 'Image') lead to higher levels of satisfaction than those requiring interaction ('SERP' and 'Visited Website'). This has important methodological consequences: we have to consider cases of *'good abandonment'*. To measure user satisfaction in this case we need to investigate the other forms of interaction signals that are not based on clicks, such as touch or swipe interactions.

### 4.5.3 Analyzing Structured Search Dialogues

We now investigate our **RQ 4.5:** *How does query-level satisfaction relate to overall user satisfaction for the structured search dialogue scenario?* Structured search dialogues are complex interactions with a longer session and different sub-

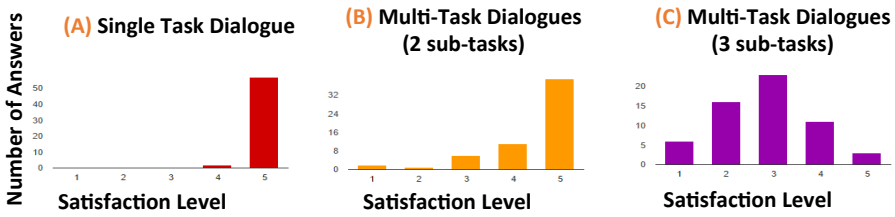



Figure 4.7: A distribution of overall user satisfaction for different types of tasks: single task search dialogues, and multi-task search dialogues with two and three objectives.

Table 4.3: Correlations of overall task user satisfaction and different summations over sub-tasks satisfaction. All presented results are statistical significant ($p < 0.05$)

| **Measures** | **Multi-Task Dialogues** |
|---|---|
| Overall SAT vs. *Average* Sub-task SAT | 0.50 |
| Overall SAT vs. *Minimum* Sub-task SAT | 0.69 |
| Overall SAT vs. *Maximum* Sub-task SAT | 0.71 |

tasks and changes of focus within the same context. This is very different from traditional search in the query-response paradigm, and session context becomes of crucial importance.

We start our analysis of the collected user interactions with structured search dialogues by introducing the satisfaction distribution for the different types of tasks presented in Figure 4.7. We see that users are more satisfied with the single task search dialogue (A), where almost all participants give the highest possible rating. The multi-task search dialogues (B and C), that are more complex have a less skewed satisfaction distribution. This immediately shows the complexity of context in structured search dialogues: when viewed independently the quality of the results is comparable for each step of the interaction, and the high levels of satisfaction for the single task search dialogue confirm that the quality is high, yet the satisfaction levels go down considerably when tasks are of increasing complexity. This suggests that the intelligent assistant loses context of a conversation, and requires more effort and interaction to restart the dialogue and get back on track. This observation is in line with our previous finding that the amount of effort users spend on a task is a principal component of user satisfaction.

We look now in greater detail at the multi-task search dialogues that contain 2 or more sub-tasks, and try to find out how overall user satisfaction is related to user satisfaction per sub-task. Table 4.3 presents the correlation between the overall *task*-level satisfaction and the minimum, mean, and maximum *query*-level

satisfaction per sub-task. The results suggest that overall user satisfaction with the complex dialogues depends more on either user frustration—some sub-tasks result in low satisfaction and frustration dragging down the overall satisfaction fast—or on user success—high levels of satisfaction with the main sub-task solving the problem lead to high levels of overall satisfaction. This has important methodological consequences: user satisfaction with the structured search dialogues cannot be measured by averaging over satisfaction with sub-tasks, suggesting that task-level satisfaction is different from sub-task or query-level satisfaction, and session-level features are a crucial component.

To summarize, this section showed the main results of the user study. We first looked at user satisfaction and found high levels of satisfaction throughout, but important differences between the scenarios on the factors contributing to overall satisfaction: the device control scenario completion correlates well with user satisfaction—it either worked or it did not—but the informational scenarios effort has a much higher correlation with user satisfaction than completion. We then looked in detail at the web search scenario. We found satisfaction dropping fast with the number of issued queries. We also found that direct answers (not requiring interaction) had higher levels of user satisfaction than SERP or webpage results (requiring further interaction) making 'good abandonment' a frequent case and necessitating to take other features (e. g. touch, swipe, acoustic) into account to discern good and bad abandonment. Finally, we zoomed in on the structured search dialogues and found high level of satisfaction per sub-task, but a drop in overall satisfaction for multi-task search dialogues with multiple subtasks addressing different aspects, showing the importance of preserving session context and demonstrating that task-level satisfaction cannot be reduced to query- or impression-level satisfaction.

## 4.6 Conclusions

This chapter aimed to answer the following main research question: *What determines user satisfaction with intelligent assistants?*, by investigating key aspects that determine user satisfaction for different scenarios of intelligent assistant usage. Our first research question was: **RQ 4.1:** *What are characteristic types of scenarios of use?* We proposed three main types of scenarios of use: **(1)** device control; **(2)** web search; and **(3)** structured search dialogue. The scenarios were identified on the basis of three factors: their proportional existence in the logs of a commercial intelligent assistant; the way requests are handled at the intelligent assistant backend (e. g. user requests are redirected to the different services and they serve different interfaces); and the way scenarios were defined in previous works [123]. Next, we investigated: **RQ 4.2:** *How can we measure different aspects of user satisfaction?* We designed a series of user studies tailored to the three scenarios of use, with questionnaires on variables potentially related to user satisfaction. The used tasks were based on an extensive analysis of logs of a commercial intelligent assistant.

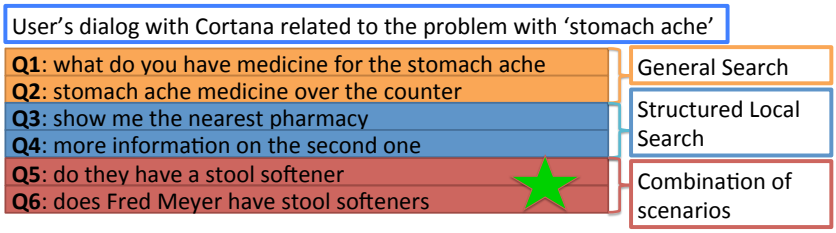



Figure 4.8: Example of a mixed dialogue.

The data collected in the user study was used to investigate the remaining research questions. First, we looked at: **RQ 4.3:** *What are key factors determining user satisfaction for the different scenarios?* We collected participant's responses on their satisfaction with the task, their ability to complete a task, and the estimated effort it took. Our main conclusion is that effort is a key component of user satisfaction across the different intelligent assistants scenarios. Second, we focused on the web search interactions: **RQ 4.4:** *How to characterize 'abandonment' in the web search scenario?* We clearly demonstrated a 'presence' of 'good abandonment' in the web search scenario, and concluded that to measure user satisfaction we need to investigate the other forms of interaction signals that are not based on clicks or reformulation. Third, we zoomed in on the structured dialogue interactions: **RQ 4.5:** *How does query-level satisfaction relate to overall user satisfaction for the structured search dialogue scenario?* We looked at user satisfaction as 'a user journey towards an information goal where each step is important,' and showed the importance of session context on user satisfaction. Our experimental results show that user satisfaction cannot be measured by averaging over satisfaction with sub-tasks. Hence, frustration with some steps in a user's 'journey' can greatly affect their overall satisfaction.

Our general conclusion is that the factors contributing to overall satisfaction with a task are different between the scenarios. Task completion is highly correlated with user satisfaction for the device control scenario—it either worked or it did not. For information seeking scenarios, user satisfaction is more related to effort than task completion. We demonstrated that task-level satisfaction cannot be reduced to query or impression-level satisfaction for information seeking scenarios.

Research on intelligent assistants for mobile devices is a new area, and this chapter addresses some of the important first steps. This work can be extended in two main directions. First, our taxonomy of three types of scenarios could be extended in various ways. In the logs we noticed that users use a mix of scenarios in order to satisfy their information needs. Consider for example the dialogue in Figure 4.8, in which the user combined multiple different scenarios in order to accomplish his/her task: The user started by using general web search (Step 1: $Q_1 \rightarrow Q_2$) to get information about his/her problem. Then he/she used the structured search dialogue (Step 2: $Q_3 \rightarrow Q_4$) to find a pharmacy. Afterwards, he/she attempted to combine the information from the prior steps through com-

plex requests (Step 3: $Q_5 \rightarrow Q_6$). Unfortunately, this led to dissatisfaction as the intelligent assistant failed to process Step 3. Therefore, it is essential to study user satisfaction when users use a mix of scenarios. Second, we found that typical behavioral signals in interaction logs (e. g. clicks) are not sufficient to infer user satisfaction with intelligent assistants. Going forward, therefore, it will be important to make use of other types of interactions such as touch or swipe, or acoustic signals to predict user satisfaction. It has been shown [89, 123, 159] that these signals are promising to detect user satisfaction with intelligent assistants and hold the potential to construct accurate predictions of task-level user-satisfaction based on behavioral data. Ultimately, such signals can be used in production systems to improve the quality of human interaction with intelligent assistants.

# 5 Search Dialogues

There is a rapid growth in the use of voice-controlled intelligent personal assistants on mobile devices, such as Microsoft's Cortana, Google Now, and Apple's Siri. They significantly change the way users interact with search systems, not only because of the voice control use and touch gestures, but also due to the dialogue-style nature of the interactions and their ability to preserve context across different queries. Predicting success and failure of such search dialogues is a new problem, and an important one for evaluating and further improving intelligent assistants. While clicks in web search have been extensively used to infer user satisfaction, their significance in search dialogues is lower due to the partial replacement of clicks with voice control, direct and voice answers, and touch gestures.

In this chapter, we propose an automatic method to predict user satisfaction with intelligent assistants that exploits all the interaction signals, including voice commands and physical touch gestures on the device. First, we conduct an extensive user study to measure user satisfaction with intelligent assistants, and simultaneously we record all user interactions. Second, we show that the dialogue style of interaction makes it necessary to evaluate the user experience at the overall task level as opposed to the query level. Third, we train a model to predict user satisfaction, and find that interaction signals that capture the user reading patterns have a high impact: when including all available interaction signals, we are able to improve the prediction accuracy of user satisfaction from 71% to 81% over a baseline that utilizes only click and query features.

## 5.1 Introduction

Spoken dialogue systems have been thoroughly studied in the literature [176, 228–230]. However, it has only been in recent years that a new generation of intelligent assistants, powered by voice, such as Apple's Siri, Microsoft's Cortana, Google Now, etc. have become a common feature on mobile devices. A recent study [81], executed by Northstar Research and commissioned by Google, found out that 55% of the U.S. teens use voice search every day and that 89% of teens

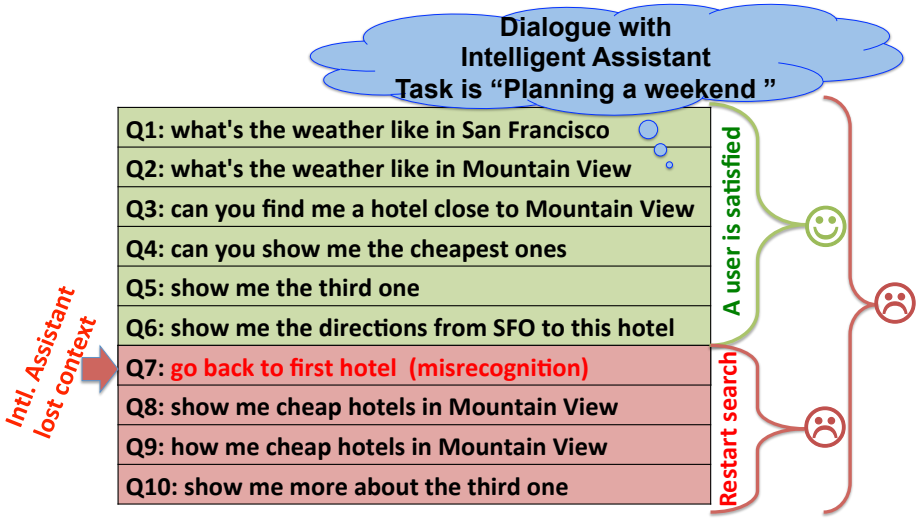



Figure 5.1: Example of search dialogue with intelligent assistant.

and 85% of adults agree that voice search is going to be 'very common' in the near future. One of the reasons for the increased adoption is the recent significant improvement in accuracy of automatic speech recognition [178]. Intelligent assistants support multiple scenarios ranging from web search to proactive user recommendations [211]. In this work, we focus on dialogue mode of interaction with intelligent assistants. In this mode, a conversation takes place between the user and the intelligent assistant: the user speaks to the intelligent assistant, it responds and the user speaks back, frequently referring to the subject of the previous request. This method of interaction is a more natural way for people to communicate and is often faster and more convenient (e. g., while driving) than typing. We call this type of interaction with intelligent assistants —*search dialogue*.

In search dialogue, users go through a sequence of steps in order to reach a desired goal: they solve one or more tasks, each of which consists of one or more search queries. As an example, consider the user dialogue in Figure 5.1: our user is trying to arrange a weekend in San Francisco. She has many tasks, from checking the weather to finding a hotel, or finding directions, etc. The user is engaged in a 'true' dialogue, i. e. the context is carried over across queries. When the intelligent assistant loses this context on $Q_7$, the user has to repeat some of the queries to rebuild the context and most probably gets dissatisfied with the intelligent assistant. So search dialogues are complex interactions, powered by voice control, with longer sessions consisting of different tasks and changes of focus within the same context. This is very different from traditional search in the query-response paradigm, and here session context becomes of crucial importance.

Clearly, evaluation of user satisfaction is an essential part of the development

of any intelligent assistant, as well as any traditional web search application. The ability to measure user satisfaction provides an understanding of the direction to take in order to improve the system. We can see from the example in Figure 5.1 that user satisfaction with search on intelligent assistants makes sense only for the entire dialogue, not as satisfaction with each query of a dialogue separately.

This prompts the need to better understand how users interact with search dialogue and how to define success and failure in terms of user experience—when are users (dis)satisfied? More specifically, we want to understand how we can measure and predict user satisfaction with search dialogue in ways that reflect perceived user satisfaction, and whether we can use traditional methods of offline and online evaluation or need to take other factors into consideration. The common practice for evaluating is to create a 'gold' standard (set of 'correct' answers) judged by editorial judges [119]. In the case of search dialogue, there may be no general 'correct' answer since the answers are highly personalized and contextualized (e. g. to a user's location or a user's past searches) to match better user-information needs. Another way to evaluate web search performance is through the use of implicit relevance feedback such as clicks, query length and landing page dwell time [11, 77, 97, 124, 125].

User satisfaction is widely adopted as a subjective measure of the quality of the search experience [131]. We know that user satisfaction for mobile web search is already very different when compared to desktop search [159] . The case of search dialogue is even more challenging for the measurement of user satisfaction [123]. Due to voice input-output to obtain answers directly from search dialogue without clicking, implicit relevance signals become far more important The use of voice commands leads to a substantial increase in the length of queries: from 3.26 terms per query on average for mobile search to 4.48 for search dialogue, while also dramatically lowering the number of clicks per Search Engine Result Page (SERP): from 0.67 to 0.30.[1] Previous work [123] has modeled user satisfaction with intelligent assistants using generic explicit interaction signals (e. g. clicks, intelligent assistants request and response features, etc.) to simulate mobile search tasks, but the characteristics of more complex interactions and important touch-based signals were left unexplored. In this chapter, we encompass all touch-based physical gestures that control the *mobile viewport location* (visible region on the mobile device), and screen taps (clicks), for the purpose of inferring user satisfaction with search dialogue. Concretely, our main research problem is:

> *How can we automatically predict user satisfaction with search dialogues on intelligent assistants using click, touch, and voice interactions?*

We break down our general research problem into three specific research questions.

**RQ 5.1:** *How can we define user satisfaction with search dialogues?*

[1] Statistics are calculated based on two weeks traffic of a commercial intelligent assistant in July 2015.

As we show in Figure 5.1, a search dialogue is a sequence of user queries where each query is a step towards user satisfaction or frustration. We analyze interactions within search dialogue, gradually increasing complexity of tasks and looking at satisfaction with tasks.

> **RQ 5.2:** *How can we predict user satisfaction with search dialogues using interaction signals?*

Clicks in web search have been extensively used to infer user satisfaction, but clicks in search dialogue have lower significance due to the use of voice control and direct answers that do not require users to click. More insights can be gained by considering other interaction signals that characterize physical interaction with mobile devices. We investigate whether users' touch interactions provide useful signals for modeling user satisfaction for search dialogue and if they are more effective than using general query, session, and click-based features.

> **RQ 5.3:** *Which interaction signals have the highest impact on predicting user satisfaction with search dialogues?*

While training an interaction-based predictor of satisfaction for search dialogue, we analyze if touch-based features are important. Furthermore, we investigate which interaction signals are more important to predict user satisfaction by performing a correlation analysis between the interaction features. To answer our research questions, we set up a lab study with realistic tasks [38] for search dialogue derived from real user logs of a commercial intelligent assistant, measuring a wide range of aspects of user satisfaction. We use the outcome of the user study to understand and predict user satisfaction with intelligent assistants.

The remainder of this chapter is organized as follows. Section 5.2 describes earlier work and background. We define user satisfaction though interaction signals for search dialogues in Section 5.3. Then, Section 5.4 introduces an approach for modeling user interaction with search dialogues. Section 5.5 provides a detailed description of the user study design to gather satisfaction labels. Finally, Section 5.6 reports our results, findings, and limitations. We conclude and discuss possible extensions of the current work in Section 5.7.

## 5.2 Background and Related Work

This chapter is relevant to three broad strands of research. First, we discuss research on spoken dialogue systems which are predecessors of the current intelligent assistants on mobile devices. Section 5.2.1 will extend our discussion of spoken dialogue systems from Section 4.2.1. Second, our work is related to evaluation of search quality because we propose a new model to evaluate user satisfaction with search dialogues (Section 5.2.2). Third, our work is closely connected to the previous studies about user satisfaction in web search systems because we suggest a way to define and predict user satisfaction for search dialogues. Section 5.2.3 supplements our discussion about user satisfaction started in Section 4.2.2 and Section 4.2.3.

### 5.2.1 Spoken Dialogue Systems

The main difference between traditional web search and intelligent assistants is their conversational nature of interaction. In the conversation mode of intelligent assistant, the technology can refer to the previous users' requests in order to understand the context of a conversation. For instance, in Figure 5.1 by asking $Q_4$ the user assumes that the intelligent assistant will 'know' that she is still interested in 'hotels in Mountain View'. Therefore, spoken dialogue systems [176] are closely related to intelligent assistants. Spoken dialogue systems understand and respond to the voice commands in a dialogue form; this area has been studied extensively over the past two decades [228–230]. Most of these studies focused on systems that have not been deployed in a large scale and hence did not have the necessary means to study how users interact with these systems in real-world scenarios, which led to most of the effort in evaluating spoken dialogue systems focusing on offline evaluation. Moreover, intelligent assistants on mobile devices support multiple scenarios of use compared with traditional spoken dialogue systems. For example, in addition to voice system response, intelligent assistants on mobile devices provide web search results, direct answers or proactive recommendations [211]. From these perspectives, intelligent assistants are similar to multi-modal conversational systems [101, 236].

This work is different from previous work on spoken dialogue systems in a way that, we study intelligent assistants on mobile devices and focus on analysis of user behavior that allows us to evaluate the system in an online setting, as well as to identify instances of dissatisfaction with the system performance.

### 5.2.2 Search Quality Evaluation

Historically, the key objective of information retrieval systems is to retrieve relevant information, typically in the form of documents or references to documents [196, 199]. In the simplest form, relevance can be defined as a score for a query-document pair. Given this query-document relevance score, many metrics have been defined, such as MAP, NDCG, DCG, MRR, P@n, TBG, etc. [119]. For such a setup, we have a collection of documents and queries that are annotated by human judges; such a setup is commonly used at TREC[2].

Recently online controlled experiments, such as A/B testing, have become widely used a technique for controlling and improving search quality based on data-driven decisions [155]. This methodology has been adopted by many leading search companies such as Bing [62], Google [223], Facebook [20], and Yandex [71]. An A/B test is designed to compare two variants of a method (e. g. ranking on SERP, ads ranking at the same time by exposing them to two user groups and by measuring the difference between them in terms of a *key metric* (e. g. the revenue, the number of visits, etc.), also known as an overall evaluation criterion. There are many existing studies towards better online evaluation which were devoted to inventing new metrics [70, 73] or improving existing ones [71]. The main goal of these studies was to make these metrics more consistent with

[2]Text REtrieval Conference: `http://trec.nist.gov/`

the long-term goals [155]. User engagement metrics show different aspects of user experience. For instance, they can reflect **(1)** *user loyalty* – the number of sessions per user [216], **(2)** *user activity* – the number of visited web pages [165] or the absence time [73]. The periodicity engagement metrics of user behavior, which resulted from the Discrete Fourier transform of state-of-the-art engagement measures were applied in [70].

Our work is related to the online evaluation line of work since our objective is to build models that can be used to evaluate intelligent assistants, possibly in A/B testing settings. Our work is different in the way that we do not focus on how to run A/B experiments, but we only focus on creating models that can be used to predict satisfaction.

### 5.2.3 User Satisfaction

User satisfaction is widely adopted as a subjective measure of search experience. Kelly [131] proposes a definition: *'satisfaction can be understood as the fulfillment of a specified desire or goal'*. Furthermore, recently researchers studied different metrics reflective of user satisfaction, such as effort [249], and it has been shown that user satisfaction at the query-level can change over time [145, 148] due to some external influence. These changes lead to the necessity of updating the data collection. *Query-level* satisfaction metrics ignore the information about users' 'journey' from a question to an answer which might take more than one query [120]. Al-Maskari et al. [14] claim that *query-level* satisfaction is not applicable for informational queries. Users can run follow-up queries if they are unsatisfied with the returned results; reformulations can lead users to an answer – this scenario is called *task-level* user satisfaction [66, 97]. Moreover, Kelly et al. [133] have provided evidence that the most complex search tasks were similar to the work [41] characterization of complex tasks with respect to having multiple interdependent parts that needed to be addressed separately.

Previous research proposed different methods for identifying successful sessions. Hassan et al. [97] used a Markov model to predict success at the end of the task. Ageev et al. [9] exploited an expertise-dependent difference in search behavior by using a Conditional Random Fields model to predict a search success. Authors used a game-like strategy for collecting annotated data by asking participants to find answers to non-trivial questions using web search. On the other hand, situations when users are frustrated have also been studied. Feild et al. [75] proposed a method for understanding user frustration. Hassan et al. [98] and Hassan Awadallah et al. [99] have found that high similarity of successive queries is an indicator of an unsuccessful task. Our work is different from this line of work in the way that we focus on intelligent assistants while all these methods focus on analyzing user behavior when users interact with traditional search systems.

Most recently, user satisfaction for intelligent assistants on mobile devices started to gain attention [123]. Jiang et al. [123] focused on simulated tasks for device control and web search, and identify satisfactory and unsatisfactory sessions based on features used in predicting satisfaction on the web, as well as acoustic features of the spoken request. They do not focus on complex search

dialogues and use use generic signals commonly used in Web search satisfaction modeling such as clicks and queries.

Sometimes, the information displayed on a SERP is sufficient to satisfy the users' information need. This phenomenon is called *good abandonment* [166, 217] and was studied in [90] for mobile devices. The authors modeled viewing behavior based on touch interaction, and demonstrated the correlation of document relevance and *viewport* changing patterns on touch-enabled mobile devices. Recent research by Lagun et al. [159] extended this line of research to model the viewport for inferring user attention and satisfaction with SERPs. The absence of clicks is an emerging problem for intelligent assistants as well, because they are frequently controlled by voice input.

Wildemuth et al. [242] reviewed over a hundred interactive information retrieval studies in terms of task complexity and difficulty, and found that the number of tasks and the number of facets were the main dimensions of task complexity. Recently, Kelly [132] linked perceived task complexity and effort, suggesting that user satisfaction may depend on the amount of effort to complete a complex task.

Our work focuses on modeling user satisfaction for intelligent assistants. We specifically focus on complex interactions—search dialogue. We show that interaction signals are essential to infer user satisfaction with search dialogue and demonstrate how they can be used in practice. We also focus on studying new interaction signals (such as touch and viewport changes) to model user's attention. We introduce a general notion of user satisfaction and exploit an extended list of interaction signals in order to predict user satisfaction with search dialogue

To summarize, the key distinctions of our work compared to previous efforts are: we studied a new method of user interaction with intelligent assistants on mobile devices, search dialogue, and we proposed a method to measure and predict user satisfaction for search dialogue using touch-interaction signals. Our metric is applicable to evaluation both online (e. g., introducing a new ranker or answer type for the intelligent assistant) and offline (e. g., mining search dialogues where users are dissatisfied).

## 5.3 Defining User Satisfaction

In this section we investigate **RQ 5.1:** *How can we define user satisfaction with search dialogues?* In the case of search dialogue, the key distinction of this scenario is the ability of the intelligent assistant to maintain the context of the conversation. Moreover, responses provided by intelligent assistants can be either in the form of a structured answer or in the form of the usual mobile SERP. Figure 5.2 (A) and (B) illustrate examples of structured answers from a commercial intelligent assistant. Examples of tasks, when this type of interaction is activated, include requests about restaurants, hotels, travel, weather, etc. Structured answers differ significantly from the usual mobile SERP (e. g., Figure 5.2 (C)). We characterize different types of search dialogues based on our broad analysis of the logs from a commercial intelligent assistant, Section 5.3.1 supplements our discussion search dialogue types started in Section 4.3.3. We also present a generalized definition

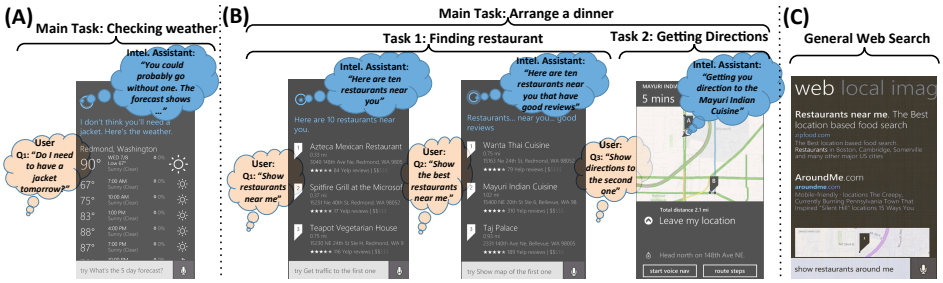



Figure 5.2: Examples (A) and (B) represent different types of intelligent assistant's response structured single task search dialogue and structured multi-task search dialogue accordingly. Example (C) represents a general SERP on mobile device.

of user satisfaction with search dialogue using interaction signals in Section 5.3.2.

### 5.3.1 Search Dialogue Types

After intensive analysis of the logs of a commercial intelligent assistant, we split search dialogues into two types: single task search dialogues and multi-task search dialogues. Roughly 50-55% of interactions can be characterized as single task search dialogues, the rest as multi-task search dialogues.[3]

**Single Task Search Dialogue** Single task search dialogue has one underlying atomic information need and mostly consists of one query and one answer. An example of a single task search dialogue is the weather-related information need, shown in Figure 5.2 (A). Single task search dialogues are very similar to mobile web search and follow the query-response paradigm. We expect that they can be evaluated using *query-level* satisfaction.

**Multi-Task Search Dialogue** Multi-task search dialogue consists of multiple interactions with the intelligent assistant that lead towards one final goal e. g. 'plan a night out'. These long and complex interactions can be divided into a series of tasks. Obviously, multi-task search dialogues are more complex than other search dialogues because of a greater number of interactions whereby the user speaks to the intelligent assistant, the intelligent assistant responds, the user speaks back to it and so on.

An example of multi-task search dialogue is presented in Figure 5.2 (B): the intelligent assistant is used to arrange a dinner. The user makes the following transitions in this search dialogue:

- $Q_1$ : asking for a list of the nearest restaurants.
- $Q_2$ : sorting the returned list to find the best restaurants (***During the transition*** $Q_1 \rightarrow Q_2$, *the intelligent assistant 'knows' that the user is referring*

[3]The terminology to characterize search dialogue types, which was introduced in Section 4.3.3, is changed here as new terminology suits better for research questions of this chapter.

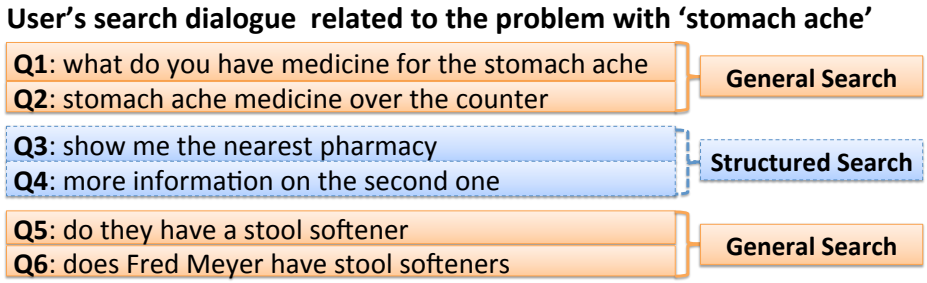



Figure 5.3: An example of the search dialogue where structured answer and general web SERP are used.

*to the list of restaurants from the previous query).*

- $Q_3$ : selecting a restaurant from the list and asking for the directions *(**During the transition** $Q_2 \rightarrow Q_3$ **:** the intelligent assistant 'knows' that the user is working with the sorted list of restaurants).*

We notice that some user needs turn out to be too complex to answer with the structured interface. An example where a user needs help with a stomach ache that is shown in Figure 5.3. In this case, the intelligent assistant used both general web search and structured dialogue interface to respond to the user's requests. The intelligent assistant redirects a user to general search if the intelligent assistant deems that general SERP will satisfy the user's information needs better such as for queries: $Q_1$, $Q_2$, $Q_5$ and $Q_6$ in Figure 5.3.

A search dialogue is not just a sequence of $\langle Q, SERP \rangle$ pairs consisting of the SERP returned by the intelligent assistant in response to the voice query $Q$. Search dialogue consists of one or more tasks, each of which consists of one or more queries. To better understand requirements for the user study setup, we divide search dialogues into single- and multi-task. Our hypothesis is that it is important for evaluation of user satisfaction with intelligent assistants if a response to a voice query $Q$ can be either in a structured form ($SERP^{str}$, see Figure 5.2 (A) and (B)) or in a form of a general web search ($SERP^{web}$, see Figure 5.2 (C)).

### 5.3.2 User Satisfaction with Search Dialogues

Based on our analysis of a commercial intelligent assistant logs[4] we hypothesize that much of the frustration happens when the intelligent assistant is not able to maintain the context and users need to start their search over in order to complete their tasks. As we present in the example in Figure 5.1, the intelligent assistant lost the context in the transition $Q_6 \rightarrow Q_7$ due to an automatic speech recognition error, and the user had to start over. Overall user satisfaction with the search

[4]We used logs of Microsoft Cortana.

dialogue decreases dramatically in this case despite the fact that the user seemed to be satisfied with the previous transitions: $Q_1 \rightarrow \cdots \rightarrow Q_6$. Furthermore, it is likely to be especially frustrating since the mistake happens after the context was transferred between the previous six queries (at the end of the session).

Single task search dialogue has one main task $T$ that can be represented as follows: $T = \big(\langle Q_1, SERP_1\rangle, \ldots, \langle Q_n, SERP_n\rangle\big)$. For any given task $T$, there is a set of interaction signals (e. g. touch, viewport change, etc.) that we denote as $I(T)$ and it can be defined as function $f$ that combines all interactions for every $\langle Q, SERP\rangle$ pair in $T$:

$$I(T) = f\Big(I\big(\langle Q_1, SERP_1\rangle\big), \ldots, I\big(\langle Q_n, SERP_n\rangle\big)\Big). \tag{5.1}$$

In the case of multi-task search dialogue, the search dialogue has more than one task and can be viewed as a sequence of tasks: $T_1, \ldots, T_m$. Interaction signals within the search dialogue are defined through the function $g$ that aggregates user interaction over tasks happening during search dialogue:

$$I(T_1, \ldots, T_m) = g\big(I(T_1), \ldots, I(T_m)\big). \tag{5.2}$$

Our objective is to define a function $h$ that given a set of interaction signals would predict whether the user was satisfied or not. For multi-task search dialogues, $h$ can be defined as:

$$SAT(T_1, \ldots, T_m) = h\big(I(T_1, \ldots, T_m)\big). \tag{5.3}$$

In the case of a single task search dialogue that consists of one task $T$, Equation 5.3 would be simplified to $SAT(T) = h\big(I(T)\big)$. If a single task search dialogue consists of a single query, the equation can be further simplified to $SAT(T) = h\big(I(\langle Q, SERP\rangle)\big)$, like in standard query-level satisfaction.

In this section, based on extensive analysis of the logs of an intelligent assistant, we characterized search dialogues as single- and multi-task, divided queries as giving either a structured or a general web search response, and conceptually modeled user satisfaction with search dialogues. Additionally we illustrated that the overall user satisfaction with search dialogue cannot be reduced to the query or even task level satisfaction, because of the dependency between them and the expectation that the intelligent assistant maintains the context during the whole interaction within a dialogue.

## 5.4 Modeling User Interactions

This section addresses **RQ 5.2:** *How can we predict user satisfaction with search dialogues using interaction signals?* First, we describe used interaction signals that are logged as the following two types of features: (1) general *implicit features* which have been used in previous work on characterizing user behavior with general Web search [11, 77, 97, 124] and intelligent assistants [123] (Section 5.4.1),

Table 5.1: Description of implicit features per search dialogue

| Feature Name | Feature Description |
|---|---|
| $F_1$ *NumQueries* | Number of queries |
| $F_2$ *NumClicks* | Number of clicks |
| $F_3$ *NumSATClicks* | Number of clicks ($>$ 30 sec. dwell time) |
| $F_4$ *NumDSATClicks* | Number of clicks ($\leq$ 15 sec. dwell time) |
| $F_5$ *TimeToFirstClick* | Time (seconds) until the first click |
| $F_6$ *MetaphoneLevenstein* | Levenstein similarity between pronunciation and writing |
| $F_7$ *MetaphoneSubstring* | Substring similarity between pronunciation and writing |

and (2) *touch and attention features* which, we believe, provide a different perspective for modeling satisfaction with search dialogues (Section 5.4.2). Note that some of these features were also shown to be useful for predicting the relevance of web search results [89, 90, 159]. These two types of features are used to define $I(\langle Q, SERP \rangle)$ which is a component of Equation 5.1. Finally, we present a method for modelling user interaction with the search dialogue task $T$ to represent $I(T)$ from Equation 5.1 (Section 5.4.3).

### 5.4.1 Query, Session and Voice Features

Table 5.1 lists the utilized implicit features: $(F_1, \ldots, F_7)$.

**Queries and Click Features** $(F_1, \ldots, F_5)$**:** In our case *click* means tapping a result item (e. g. the best answer from a list of candidates). We use the following features that are calculated across the entire search dialogue task: the number of queries ($F_1$) the number of clicks ($F_2$), the number of satisfied clicks, defined as clicks with dwell time $>$ 30 seconds ($F_3$), as well as the number of dissatisfied clicks, defined as clicks with dwell time $\leq$ 15 seconds ($F_4$), and the total time (seconds) before the first click in search dialogue ($F_5$). Note that previous work [77] has shown that long dwell time clicks ($>$ 30 seconds) are highly likely to indicate satisfaction while quick-back clicks ($\leq$ 15 seconds) are highly likely to indicate dissatisfaction.

**Acoustic Features** $(F_6, F_7)$**:** We utilize acoustic features to characterize voice interaction happening in search dialogues. More specifically, we use the phonetic similarity between consecutive requests to identify patterns of repetition. Metaphone representation [182] is a way of indexing words by their pronunciation that allows us to represent words by how they are pronounced as opposed to how they are written. Phonetic similarity is assessed by computing the edit distance between the Metaphone representation of two utterances. For example, a voice query *'WhatsApp'* may be incorrectly recognized as *'what's up'*, but their metaphone codes are both *'WTSP'*. In such cases, this phonetic similarity feature helps us detect repeated or similar requests that are missed by normal text similarity

Table 5.2: Description of touch features per search dialogue

| Feature Name | Feature Description |
|---|---|
| $F_9$ *NumSwipes* | Number of Swipes |
| $F_{10}$ *NumUpSwipes* | Number of up-swipes |
| $F_{11}$ *NumDownSwipes* | Number of down-swipes |
| $F_{12}$ *SwipedDistance* | Total distance swiped (pixels) |
| $F_{13}$ *AvgNumSwipes* | Number of swipes normalized by time |
| $F_{14}$ *AvgSwipeDistance* | Total distance divided by number of swipes |
| $F_{15}$ *DistanceByTime* | Total swiped distance divided by time |
| $F_{16}$ *DirectionChanges* | Number of swipe direction changes |
| $F_{17}$ *DurationPerAns* | SERP answer duration (seconds) which is shown on screen (even partially) |
| $F_{18}$ *FractionPerAns* | Fraction of visible pixels belonging to SERP answer |
| $F_{19}$ *ReadTimePerAns* | Attributed time (seconds) to viewing a particular element (answer) on SERP |
| $F_{20}$ *1DReadTimePerPix* | Attributed time (seconds) per unit height (pixels) associated with a particular element on SERP |
| $F_{21}$ *2DReadTimePerPix* | Attributed time (milliseconds) per unit area (square pixels) associated with a particular element on SERP |

features based on recognized speech. As similarities metrics we use Levenstein Distance ($F_6$) and Substring ($F_7$).

### 5.4.2 Touch Features

One of the main contributions of this work is the introduction of touch and attention features for detecting user satisfaction with search dialogue. We focus on touch-based features related to the way in which users interact with the screen and features based on elements visible to users. This serves as a surrogate for what the user is paying attention to on the page and how this changes throughout the search dialogue. Table 5.2 lists the utilized touch features.

Capturing touch events is not easy in practice because of non-standard instrumentation [107]. We derive interaction features and the exact information that was displayed on the phone screen at any given time using *mobile viewport logging*. This allows us to record the portion of the answer/result currently visible on the screen, as well as bounding boxes of all results shown on the page. For instance, if an element is visible in the viewport at some point in time and then no longer visible, one can infer that a gesture must have taken place. Furthermore, if an element below the original element becomes visible, then one can infer that it must have been a downward swipe action. We use element-tracking in the viewport to infer features related to swipes happening during search dialogue: $F_9, \ldots, F_{16}$.

Lagun et al. [159] showed that there is a strong correlation between the time

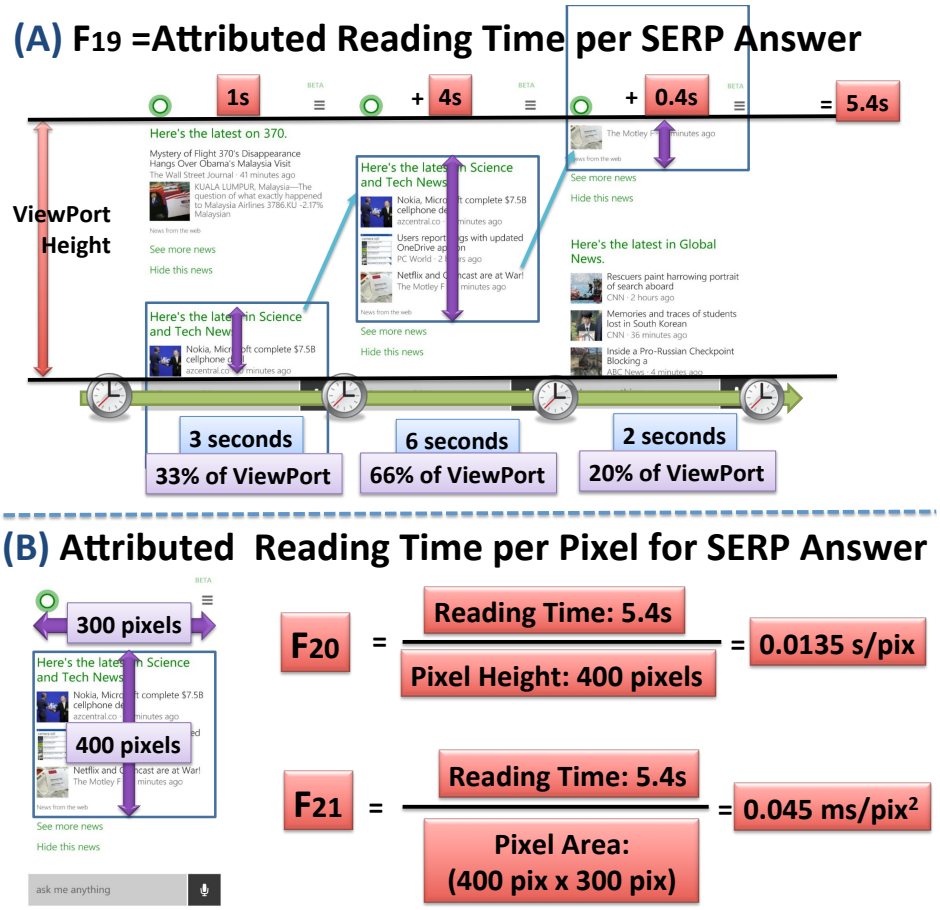



Figure 5.4: The illustration how to capture (A): $F_{19}$ *ReadingTimePerAnswer* and (B): $F_{20}$, $F_{21}$*ReadingTimePerPixel*.

for which a result is visible and its gaze time. Following this observation, we approximate how much attention different SERP elements get. Features $F_{17}, F_{18}$ are used to characterize visibility of SERP answers. The feature $F_{19}$ attempts to attribute the time the user spends looking at each stationary viewport to the different elements based on their area. Features $F_{20}$ and $F_{21}$ are responsible for reading time per pixel, they normalize the attributed reading time so that size of the content region does not introduce a systematic weight into the metric. Figure 5.4 illustrates how $F_{19}$ is captured in the example (A) and how $F_{20}$ and $F_{21}$ are calculated in the example (B). To aggregate the features $F_{17}, \ldots, F_{21}$ at the $\langle Q, SERP \rangle$-level, we use four types of aggregation: average ($Avg$), maximum ($Max$), minimum ($Min$), and standard deviation ($Stdev$).

We presented a list of implicit and touch features that are collected on $\langle Q, SERP \rangle$-

level during user interaction with search dialogues on intelligent assistants. We define $I(\langle Q, SERP\rangle)$ by the feature vector: $(F_1, \ldots, F_{21})$. Next, we will explain how to model user interaction on the task level—$I(T)$.

### 5.4.3 User Interactions over Search Dialogues

We showed that search dialogue tasks have an underlying semantic structure and potentially can be divided into single task search dialogues and multi-task search dialogues. There is no automatic search dialogue analyser available so we cannot split search dialogues into tasks on the fly. The goal of this work is to deliver an online metric for user satisfaction with search dialogues on intelligent assistants. Potentially, the semantic structure of a search dialogue task is not entirely flat and it might have a tree structure. Developing an automatic tool to mine the search dialogue structure is a promising direction for future work.

The intelligent assistant has two types of response to a voice query $Q$: either in a structured form, $SERP^{str}$ as illustrated in Figure 5.2 (A) and (B), or in the form of a general web search, $SERP^{web}$ as illustrated in Figure 5.2 (C). Our hypothesis is that the type of response of intelligent assistants can be used to approximately divide search dialogues into the different types of tasks. Our assumption relies on the internal logic of the intelligent assistant that returns $SERP^{str}$ when tasks are about different types of location (restaurants, hotels, pharmacies, shops etc.), directions to locations, or weather. If the intelligent assistant deems that information from general web is more suitable for a query then it returns $SERP^{str}$. This kind of intelligent assistant response still differs from general mobile search because it looks like a dialogue. For example, if a user voice query can be answered using the knowledge graph then, the intelligent assistant speaks the answer out aloud.

We define the function $f$ from Equation 5.1 through aggregation. Further, in our experiment we use geometric mean as aggregation. We experimented with other aggregation functions, and they yielded similar or worse performance. We apply three techniques to define $I(T)$ for the search dialogue consisting of $n$ queries in total, $m$ queries resulted in $SERP^{str}$ and $k$ queries resulted in $SERP^{web}$:

**$A_1$:** considering only interaction with $\langle Q, SERP^{str}\rangle$:

$$I(T) = \Big(\prod_{i=1}^{m} I\big(\langle Q_i, SERP_i^{str}\rangle\big)\Big)^{1/m}; \tag{5.4}$$

**$A_2$:** considering interactions with all $\langle Q, SERP\rangle$ equally:

$$I(T) = \Big(\prod_{i=1}^{n} I\big(\langle Q_i, SERP_i\rangle\big)\Big)^{1/n}; \tag{5.5}$$

**$A_3$:** separating interactions with $\langle Q, SERP^{str}\rangle$ and $\langle Q, SERP^{web}\rangle$ as two differ-

ent tasks $T^{str}$ and $T^{web}$:

$$I(T^{str}, T^{web}) = \Big[\Big(\prod_{i=1}^{m} I\big(\langle Q_i, SERP_i^{str}\rangle\big)\Big)^{1/m}, \Big(\prod_{j=1}^{k} I\big(\langle Q_j, SERP_j^{web}\rangle\big)\Big)^{1/k}\Big]. \tag{5.6}$$

In this section we introduced the list of features to model user interactions. We focused specifically on presenting interaction signals which are promising for modeling user interaction with intelligent assistants. Next, we will describe the setup for our lab study with real-world tasks for search dialogue derived from real user logs. The outcome of the study will be used to understand how important interaction signals are for modeling user satisfaction with search dialogue.

## 5.5 User Study

This section describes the design of the user study to collect user interactions for search dialogues. The more general discussion about designing user studies for intelligent assistants is initially presented in Section 4.4. The collected data is used to investigate our research questions: **RQ 5.2** and **RQ 5.3**. While designing tasks for our user study, we rely on the following requirements: (1) the suggested tasks should be realistic; (2) following Borlund [38] we construct the tasks so that participants could relate to them and they would provide '*enough imaginative context.*'

**Participants** We recruited 60 participants to participate in the study. All participants were college or graduate students residing in the United States. They all had good command of English. 75% of the participants were male and the remaining 25% were female. The average age of the participants is 25.5 ($\pm$ 5.4) years. They were reimbursed by a $10 gift card for participating in the study.

**Tasks** We analyzed over 400,000 search dialogues from the search logs of a commercial intelligent assistant to generate tasks for the user study. Based on our analysis we generated **eight** tasks for the user study that were designed to cover approximately 70-80% of subjects queried by real users of the intelligent assistant. We formulated tasks in a free form in order to encourage query diversity and stimulate either genuine satisfaction or frustration with returned results. The final tasks for the user study consist of:

- **one** single task search dialogue that is related to the weather e. g. **Task A:** *'Check if you need to a coat tomorrow?'*;
- **four** multi-task search dialogues that include two subjects e. g. **Task B:** *'You are planning a night out. Pick a restaurant based on your preferences: cheap, best review, or closest. Find out driving directions to the selected restaurant*;

- **three** multi-task search dialogues that require at least three switches within the same context e. g. **Task C:** *'You are planning a vacation. Pick a place. Check if the weather is good enough for the period you are planning the vacation. Find a hotel that suits you. Find out driving directions to this place.*

For each task, we recorded an audio that verbally described the task objective. Following the study [123], we did not show the participants the written description while they were working on the task as it was demonstrated many participants directly used the sentences shown in descriptions as requests. We strongly wanted to avoid such outcome because our goal was to simulate real user behavior.

**Study Setup** Participants performed the tasks on a mobile phone with a commercial intelligent assistant installed. If the task needed access to specific device resources, functions or applications (e. g. maps), they were pre-installed to make sure users would not encounter problems. The experiment was conducted in a quiet room, so as to reduce the disturbance of external noise. Although the real environment often involves noise and interruption, we eliminated those factors to simplify the experiment. While participants were doing the user study all their interactions were logged using an internal API.

The participants watched a 4 minutes video with instructions that explained how to use the intelligent assistant. Then, participants worked on **one** training task and **eight** formal tasks. We instructed participants that they should stop a task when they had accomplished their goal or if they became frustrated and wanted to give up. After completing each task, participants were asked to answer the following four questions:

1. *Were you able to complete the task?*
2. *How satisfied are you with your experience in this task in general?*

   In case of multi-task search dialogue participants indicate their graded satisfaction e. g. for **Task C** questions were:

   2.1 *How satisfied are you with finding a hotel?*
   2.2 *How satisfied are you with finding a direction?*
3. *Did you put in a lot of effort to complete the task?*
4. *How well did the intelligent assistant recognize your voice?*

Except for the first question which required a Yes/No answer, all questions were answered using a 5 point Likert scale. Additionally, to stimulate participants' involvement in search dialogues, we asked them to answer clarifying question(s) about task output. For example, if the task was about finding a restaurant the participant would need to indicate its name in the questionnaire. The total experiment time was about 30 minutes.

**User Study Summary** We stimulated participants' involvement by giving free form tasks. They needed to formulate their own goals for the task and it leads

Table 5.3: Measurements of prediction quality based on different subsets of features. The relative improvements compared to the baseline are provided in parentheses

| Predictor | Acc (%) | SAT (%) P | SAT (%) R | SAT (%) $F_1$ | DSAT (%) P | DSAT (%) R | DSAT (%) $F_1$ | Avg. $F_1$ (%) | AUC (%) |
|---|---|---|---|---|---|---|---|---|---|
| **Baseline**: $A_1(F_1, \ldots, F_5)$ | 70.62 | 70.72 | **92.91** | 80.31 | 70.50 | 30.37 | 42.45 | 61.38 | 61.51 |
| $\mathbf{P}_1$: $A_1(F_1, \ldots, F_7)$ | 78.53* (11.20) | 81.81* (15.68) | 85.73 (-7.73) | 83.72 (4.25) | 71.24 (1.76) | 65.55* (115.84) | 68.51* (61.37) | 76.11* (24.00) | 81.20* (32.01) |
| $\mathbf{P}_2$: $A_1(F_1, \ldots, F_{21})$ | 78.78* (11.55) | 80.98* (14.51) | 87.75 (-5.55) | 84.23 (4.88) | 74.69 (5.94) | 62.61* (106.16) | 68.12* (60.46) | 76.17* (24.10) | 83.59* (35.90) |
| $\mathbf{P}_3$: $A_2(F_1, \ldots, F_{21})$ | 80.21* (13.58) | 82.55* (16.73) | 87.99 (-5.30) | **85.18 (6.07)** | **76.28*** **(8.20)** | 66.07* (117.55) | 70.81* (66.80) | 78.00* (27.07) | 83.31* (35.44) |
| $\mathbf{P}_4$: $A_3(F_1, \ldots, F_{21})$ | **80.81*** **(14.43)** | **84.89*** **(+20.04)** | 85.42 (-8.06) | 85.15 (6.03) | 73.45 (4.18) | **72.55*** **(138.89)** | **73.00*** **(71.95)** | **79.08*** **(28.83)** | **85.62*** **(39.20)** |

to satisfaction or frustration. For example, out of 60 responses for **Task C** we extracted 46 references to unique places. As a result of free task formulation we obtained a diverse query set, characterized as follows: in total, participants perform 540 tasks that involved 2,040 queries in total of which 1,969 were unique; the average query length was 7.07. The single task search dialogue as **Task A** generated 130 queries in total, four multi-task search dialogues as **Task B** generated 685 queries, and three multi-task search dialogues as **Task C** generated 1,355 queries.

## 5.6 Results and Findings

We now investigate our **RQ 5.3:** *Which interaction signals have the highest impact on predicting user satisfaction with search dialogues?* We begin by introducing our results on the prediction quality of user satisfaction with search dialogues (Section 5.6.1). We conclude by presenting a correlation analysis between the interaction features and user satisfaction (Section 5.6.2).

### 5.6.1 Predicting User Satisfaction

The purpose of this study is to predict overall user satisfaction with search dialogues. Therefore we do not utilize graded satisfaction in this work but it would be useful for future research. In our user study, users reported overall satisfaction using a 5 point Likert scale. Due to the large difference in rating distributions between the single- and multi-task search dialogue we consider the evaluation as a binary classification problem. We divide the labeled search dialogues into binary classes: satisfied (SAT) – users provided 5 or 4; dissatisfied (DSAT) – everything else. This resulted in the following proportion of positively and negatively labeled search dialogues: SAT – 64% and DSAT – 36%.

We formulate a supervised classification problem where, given a search dialogue, the goal is to classify it to SAT or DSAT. We train Gradient Boosted De-

cision Trees (GBDT) [78] as a satisfaction predictor $h$ presented in Equation 5.3. We experiment with other classifiers (logistic regression, SVM), and they yield similar or worse performance. Hence we only report the results of GBDT.

We use 10-fold cross validation. For each training fold, we use grid search to optimize the number of leaves, tree depth, and the number of leaves required to split. We train our predictors based on different subsets of features from $(F_1, \ldots, F_{21})$. For each experiment we report the overall accuracy (Acc), average $F_1$ score (Avg. $F_1$), area under the curve (AUC); and precision (P), recall (R) and $F_1$ score ($F_1$) for SAT and DSAT separately. The results are shown in Table 5.3.

The baseline is the classifier trained on queries and click features which are aggregated over a search dialogue using Equation 5.4. We observe that the baseline is overly optimistic with a low DSAT recall (30%) and high SAT recall (93%), showing that it is effective in picking up the imbalance in SAT/DSAT distribution but far less effective in distinguishing satisfaction from dissatisfaction. We train the predictor $P_1$ on an expanded feature set, adding the Methaphone features $(F_6, F_7)$. From Table 5.3, we can see that the predictor $P_1$ shows statistically significant improvement ($p < 0.05$) in Acc, SAT P, DSAT R, DSAT $F_1$, Avg. $F_1$ and AUC when compared against the baseline. Next, we expand feature set by adding the touch signals from Table 5.2.

We use the three proposed techniques for feature aggregation over task(s) while training based on $(F_1, \ldots, F_{21})$: $A_1$ (Equation 5.4) for the predictor $P_3$, $A_2$ (Equation 5.5) for $P_4$, and $A_3$ (Equation 5.6) for $P_5$. Based on results in Table 5.3, we can infer that the predictors $P_2$, $P_3$ and $P_4$ demonstrate statistically significant improvements ($p < 0.05$) in Acc, SAT P, DSAT R, DSAT $F_1$, Avg. $F_1$ and AUC when compared against the baseline, indicating that the touch features incorporated in prediction models are fundamental to evaluation of user satisfaction with search dialogues. Also from Table 5.3, we can infer that the aggregation $A_3$ (when we separate user interactions: $SERP^{str}$ and $SERP^{web}$) is the most beneficial one, when compared against the baseline. In the next subsection, we present features analysis to characterize the relative importance of different features.

### 5.6.2 Features Analysis

To understand the impact of implicit features $(F_1, \ldots, F_7)$ from Table 5.1, we calculate the Pearson correlation between the user satisfaction label (SAT) and each feature. The results are presented in Table 5.4. Feature $F_7(Q_i, Q_{i+1})$, which indicates that the subsequent query $Q_{i+1}$ in the task contains prior query $Q_i$, is positively correlated with SAT. Expanding the query, or rather refining the query to better specify the intent, is a common user behavior and is expected to increase the probability of finding satisfactory content on the subsequent SERP. The complementary feature, $F_7(Q_{i+1}, Q_i)$, however, reflects the case where the subsequent query $Q_{i+1}$ in the task is contained within the prior query $Q_i$; this feature is negatively correlated with SAT. Speech recognition errors in $Q_i \rightarrow Q_{i+1}$ can give rise to this type of feature, and the negative correlation is expected from such transitions. Our findings are similar to the previously reported results [123]. Based on relatively high correlation between click-based features $(F_2, \ldots, F_5)$ we

Table 5.4: Pearson correlations between satisfaction (SAT) and implicit features. Results are statistically significant ($p < 0.05$)

| Feature Type | Correlation |
|---|---|
| $F_7(Q_i, Q_{i+1})$ *[MetaphoneSubstring]* | 0.45 |
| $F_4$ *[NumDSATClicks]* | 0.31 |
| $F_5$ *[TimeToFirstClick]* | 0.30 |
| $F_2$ *[NumClicks]* | 0.27 |
| $F_6$ *[MetaphoneLevenstein]* | 0.23 |
| $F_3$ *[NumSATClicks]* | 0.12 |
| $F_7(Q_{i+1}, Q_i)$ *[MetaphoneSubstring]* | -0.16 |
| $F_1$ *[NumQueries]* | -0.49 |

infer that clicks during search dialogues can be interpreted as a sign of user satisfaction. We find that the search dialogue length, in terms of $F_1$, is negatively correlated with satisfaction. Long conversations can be the result of two types of behavior: (a) multiple attempts of users to have their speech properly recognized, or (b) the loss of context by the intelligent assistant during the conversation, forcing users to restart the conversation; both of these explain the observed negative correlation.

Table 5.5 shows the results of correlation analysis for the touch features ($F_8, \ldots, F_{21}$) using aggregations $A_2$ and $A_3$. We present the top 5 positively-correlated features and the top 5 negatively correlated features. To explain the correlations, we present three hypotheses. These hypotheses are not alternatives and can all be true together:

**$\mathbf{H}_1$:** The SERP for a query is ordered by a measure of relevance as determined by the system, then additional exploration is unlikely to achieve user satisfaction, but is more likely an indication that the best-provided results (i. e. the SERP top) are insufficient to address the user intent.

**$\mathbf{H}_2$:** In the converse case of $\mathbf{H}_1$, when users find content that satisfies their intent, their likelihood of scrolling is reduced, and they dwell for an extended period on the top viewport.

**$\mathbf{H}_3$:** When users are involved in a complex task, they are dissatisfied when redirected to a general mobile SERP, as opposed to receiving an explicit structured answer from the intelligent assistant (e. g. the transition $Q_4 \rightarrow Q_5$ in Figure 5.3). Unlike $\mathbf{H}_2$, the absence of scrolling on this landing page is an indication of dissatisfaction.

The features in Table 5.5 are explained in more depth below. A large $Stdev(F_{18})$ characterizes the situation where roughly half of the available answers is observed and the other half is not. This would occur when there is minimal or no scrolling behavior, since answers at the top of the SERP are visible and the answers toward

Table 5.5: Pearson correlations between satisfaction (SAT) and touch features. Results are statistically significant ($p < 0.05$)

| **Feature Type** | **Cor.** |
|---|---|
| $A_2$ **(Eq. 5.5) for aggregating Touch Features** | |
| $Stdev(F_{18})$ *[FractionPerAns]* | 0.23 |
| $Min(F_{20})$ *[1DReadTimePerPix]* | 0.20 |
| $Stdev(F_{19})$ *[ReadTimePerAns]* | 0.19 |
| $Avg(F_{20}))$ *[1DReadTimePerPix]* | 0.19 |
| $Max(F_{20}))$ *[1DReadTimePerPix]* | 0.18 |
| ... | ... |
| $F_{10}$ *[NumUpSwipes]* | -0.10 |
| $F_9$ *[NumSwipes]* | -0.12 |
| $F_{11}$ *[NumDownSwipes]* | -0.12 |
| $F_{12}$ *[SwipedDistance]* | -0.13 |
| $F_{15}$ *[DistanceByTime]* | -0.18 |
| $A_3$ **(Eq. 5.6) for aggregating Touch Features Aggregation** | |
| $I\big(\langle Q, SERP^{str}\rangle\big)$: $Max(F_{18})$ *[FractionPerAns]* | 0.35 |
| $I\big(\langle Q, SERP^{str}\rangle\big)$: $Stdev(F_{18})$ *[FractionPerAns]* | 0.34 |
| $I\big(\langle Q, SERP^{str}\rangle\big)$: $Max(F_{19})$ *[ReadTimePerAns]* | 0.32 |
| $I\big(\langle Q, SERP^{str}\rangle\big)$: $Avg(F_{18})$ *[FractionPerAns]* | 0.31 |
| $I\big(\langle Q, SERP^{str}\rangle\big)$: $Avg(F_{19})$ *[ReadTimePerAns]* | 0.31 |
| ... | ... |
| $I\big(\langle Q, SERP^{web}\rangle\big)$: $Min(F_{20})$ *[1DReadTimePerPix]* | -0.35 |
| $I\big(\langle Q, SERP^{web}\rangle\big)$: $Stdev(F_{18})$ *[FractionPerAns]* | -0.28 |
| $I\big(\langle Q, SERP^{web}\rangle\big)$: $Min(F_{18})$ *[FractionPerAns]* | -0.32 |
| $I\big(\langle Q, SERP^{web}\rangle\big)$: $Avg(F_{18})$ *[FractionPerAns]* | -0.35 |
| $I\big(\langle Q, SERP^{web}\rangle\big)$:$Max(F_{18})$ *[FractionPerAns]* | -0.35 |

the bottom are hidden from view. $F_{20}$ is well-defined only for observable content, and when users do not scroll, this value will be identical for all items on the SERP. As such, in the absence of scrolling, $Min(F_{20})$ will be large, and therefore a positive correlation with SAT is consistent with our hypotheses. $F_{19}$, on the other hand, is well-defined for all answers, observed or not, but will be equal to zero for answers that are not observed. When there is minimal scrolling and a long dwell on the top viewport, $F_{19}$ will be positive and large for the observed answers, and zero for the unobserved content, giving rise to a large $Stdev(F_{19})$. $Avg(F_{20})$ characterizes the same behavior as $Min(F_{20})$ when users do not scroll at all, but, when users do scroll small distances, $Min(F_{20})$ would drop substantially whereas $Avg(F_{20})$ would remain relatively stable; a positive correlation with SAT is consistent with $\mathbf{H}_2$. A large $Max(F_{20})$ implies that users paused and dwelled on one portion of the page for an extended period, also consistent with $\mathbf{H}_2$.

Table 5.5 ($A_2$) shows that SAT is negatively correlated with ($F_9$,...,$F_{12}$), which describe user swipes. Swipe down, up, or both is is a sign of exploration of the result set and a negative correlation of number of swipes and swipe-distance with SAT is consistent with $\mathbf{H}_1$. $F_{15}$ provides a measure of the speed of exploration of the content. The observed negative correlation implies that fast swiping indicates dissatisfaction, and it is consistent with users who are skimming through and exploring the results without success, supporting $\mathbf{H}_1$. These results are consistent with the findings of Lagun et al. [159], who concluded that scrolling is negatively correlated with SAT.

For the aggregation $A_3$ (Equation 5.6), we separate interaction with structured answers, $I\big(\langle Q, SERP^{str}\rangle\big)$, and interaction with general mobile SERP, $I\big(\langle Q, SERP^{web}\rangle\big)$. The correlation between SAT and $F_{18}$, $F_{19}$ calculated though interaction with $SERP^{str}$ is even stronger. The same set of features calculated for interactions with $SERP^{web}$ is negatively correlated with SAT, which is consistent with $\mathbf{H}_3$. Users who are redirected to $SERP^{web}$ and do not scroll, likely land there unintentionally as a consequence of a voice-misrecognition or loss of context by the intelligent assistant. While Table 5.5 only shows the top features, the entire list of correlations for $A_3$ are consistent with the $\mathbf{H}_1$, in agreement with our previous finding for the aggregation $A_2$. Furthermore, we can see that swiping actions during interactions with $SERP^{web}$ have a higher negative correlation than with $SERP^{str}$. Here, users are plausibly frustrated and perform quick swipes through $SERP^{web}$. The above observations lead us to the following conclusion—that users expect to find answers on the SERP without any 'additional effort' (e. g. scrolling), and users are not satisfied if the intelligent assistant cannot answer their request explicitly and redirects them to a general mobile SERP. Therefore, the aggregation $A_3$ is more sensitive to DSAT, and explains why it performed better as a predictive model

Although our work shows that our method has a strong potential, there are at least two limitations that can be improved in future work. The first limitation is the collected data during user study which can be improved in terms of size and diversity. One way to do that is to monitor users as they do their normal tasks via additional instrumentation installed on their phones and prompt them to answer questions about their satisfaction. Another area of improvement is using data collected from multiple intelligent assistants. Most available intelligent assistants support search dialogues and the features we use are independent of the task subject and hence should be useful regardless of which tasks are supported by which assistants. Nevertheless, training and testing our models on data from different assistants can be very useful for proving their generality. This is particularly challenging though given the difficulty of performing third-party instrumentation on mobile devices.

To summarize, extensively experimenting with the user study data, we concluded that touch and attention based features are extremely helpful for predicting user satisfaction with intelligent assistants. Finally, we conducted feature analysis and concluded that active user interactions with the mobile device (e. g., scrolling) is a strong signal of user dissatisfaction with intelligent assistants.

## 5.7 Conclusions

The chapter extends earlier work on desktop and general mobile search [90, 123, 159] and presents the first quantitative study for user satisfaction with the modern generation of intelligent assistants. Intelligent assistants allow for radically new means of information access: making a real dialogue with a context using voice commands and touch interactions. Evaluation of user satisfaction is absolutely necessary for intelligent assistants development. As the popularity of intelligent assistants rapidly grows, a strong need for better understanding and precise evaluating of user satisfaction grows correspondingly.

Our main research question was: *How can we automatically predict user satisfaction with search dialogues on intelligent assistants using click, touch, and voice interactions?* First, we studied **RQ 5.1:** *How can we define user satisfaction with search dialogues?* We studied search dialogues by analyzing real logs of a commercial intelligent assistant and introduced two types of the dialogues: single task search dialogues and multi-task search dialogues. We also illustrated that the dialogue queries can lead to responses either in the form of a structured interface or in the form of general mobile search, when a request is 'out of scope' of the search dialogue. We defined user satisfaction with search dialogues in the generalized form, which showed understanding the nature of user satisfaction as an aggregation of satisfaction with all dialogue's tasks and not as a satisfaction with all dialogue's queries separately. The introduction of dialogue types and understanding which kinds of responses to queries exist, helped us to set up a user study and to make feature selection for answering the next research question.

Next we investigated **RQ 5.2:** *How can we predict user satisfaction with search dialogues using interaction signals?* To predict user satisfaction, we used the following kinds of interactions: clicks (or 'taps' in terms of touches on mobile platforms), other touch interactions and voice features. The baseline was predicting user satisfaction using clicks and queries features. By conducting experiments we provided empirical evidence that features derived from voice and especially from touch interactions add significant gain in accuracy over the baseline. To understand how to efficiently select features depending on different types of queries, we proposed three techniques: using only features of queries resulting in structured interface; calculating a single set of features for queries resulting in structured interface and queries resulting in general SERP; and calculating an own set of features for each group of queries resulting in structured interface and queries resulting in general SERP. We conducted analysis and showed that the third technique is the most accurate one to model user satisfaction. This technique improves accuracy from 71% to 81% over the baseline. Which features are most important for modeling user satisfaction is shown in the answer to our final research question presented next.

Finally, we analyzed the prediction quality of the classifier trained on various selections of interaction features, answering **RQ 5.3:** *Which interaction signals have the highest impact on predicting user satisfaction with search dialogues?* We conducted the extensive feature analysis. We concluded that users expect to find answers on the SERP directly without putting in any 'additional effort'

(e. g. scrolling). Our analysis showed a strong negative correlation between user satisfaction and swipe actions. Additionally, we demonstrated that users are not satisfied if the intelligent assistant cannot answer their query explicitly and redirects them to a general mobile SERP.

Our general conclusion is that touch based features dramatically improve the prediction quality of user satisfaction with search dialogue. Research on intelligent assistants on mobile devices is a new area, and this chapter addresses some of the first important and necessary steps. We proposed a method for evaluating user satisfaction with intelligent assistants which can be applied in online evaluation of ranking results, offline mining of user dissatisfaction and understanding directions for their future development.

# 6
# Good Abandonment

Web search queries for which there are no clicks are referred to as *abandoned queries* and are usually considered as leading to user dissatisfaction. However, there are many cases where a user may not click on any search result page (SERP) but still be satisfied. This scenario is referred to as *good abandonment* and presents a challenge for most approaches measuring search satisfaction, which are usually based on clicks and dwell time. The problem is exacerbated further on mobile devices where search providers try to increase the likelihood of users being satisfied directly by the SERP. This chapter proposes a solution to this problem using gesture interactions, such as reading times and touch actions, as signals for differentiating between good and bad abandonment. These signals go beyond clicks and characterize user behavior in cases where clicks are not needed to achieve satisfaction. We study different good abandonment scenarios and investigate the different elements on a SERP that may lead to good abandonment. We also present an analysis of the correlation between user gesture features and satisfaction. Finally we use this analysis to build models to automatically identify good abandonment in mobile search achieving an accuracy of 75%, which is significantly better than considering query and session signals alone. Our findings have implications for the study and application of user satisfaction in search systems.

## 6.1 Introduction

In recent years, there has been a large increase in people using their mobile phones to access the Internet, with it being reported that, in 2013, 63% of Americans used their mobile phones to go online compared to 31% in 2009 [72]. Having immediate access to mobile devices capable of searching the Web has led to important changes in the way that people use search engines. For instance, previous research has shown that search on mobile devices is often much more focused and that the query length and intents differ from traditional search [127]. It has also been found that mobile users might formulate queries in such a way so as to increase the likelihood of them being directly satisfied by the SERP [166]. In addition to

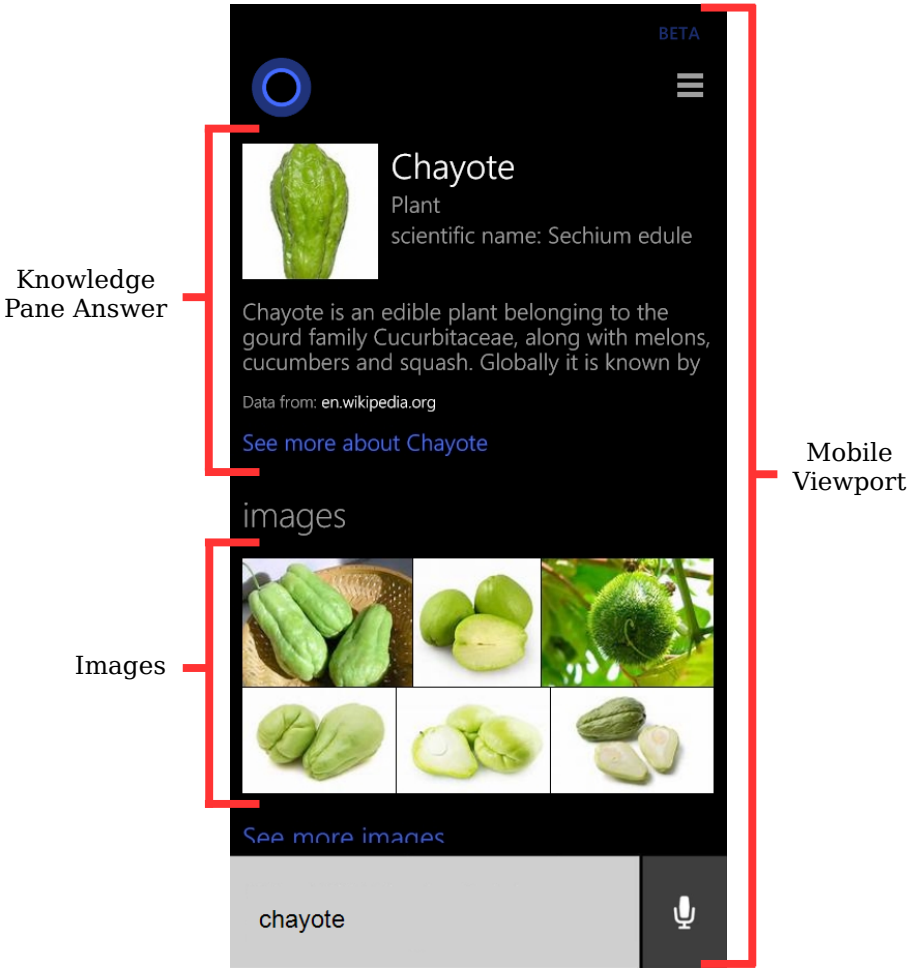



Figure 6.1: An example of a mobile SERP, showing the viewport, an answer and images.

these differences, the mobile screen sizes are typically much smaller than that of non-mobile devices. As a result of these differences, search engines have had to adapt in order to be able to better satisfy mobile users.

One way this has been done is by search engines presenting *answers* on the SERP in response to user queries. These answers typically come in the form of boxes containing a fact and, when present, they have the ability to satisfy

the user need immediately. On mobile devices, there are many times when this may occur. For instance, a user may be out with friends and needs to find the answers to questions that come up in conversation, such as *what will the weather be like tomorrow? What time does the movie start tonight?* Or *what year was a celebrity born?* Many of these types of questions can be answered by search engines without users needing to click on search results. Figure 6.1 shows an example of an answer that appears in the mobile search on Microsoft's digital assistant Cortana. The answer, which shows information about a plant, has the potential to directly satisfy the user's information need on the mobile SERP and thus may negate the need for the user to click on any hyperlinks. Furthermore, while it is clear that answers on a mobile SERP may satisfy a user, it is also possible for other elements on the SERP to do this. For instance, users can be satisfied by good snippets and images in SERPs.

Good abandonment refers to the case where a user is directly satisfied by the SERP without the need to click on any hyperlinks and the user is said to *abandon* the query [217]. This is in contrast to bad abandonment where a user abandons their query due to being dissatisfied by the search results. It has been shown that good abandonment is more likely in mobile search. For instance, a study in 2009 estimated that 36% of abandoned mobile queries in the U.S. were likely good compared to 14.3% in desktop search [166].

Traditionally, abandoned queries have been considered a bad signal when measuring the effectiveness of search engines; however, recently there has been increasing awareness that abandonment can also be a good thing [30, 54, 55, 166, 217]. However, most approaches for measuring search satisfaction and success have been based on implicit feedback signals such as clicks and dwell time [77, 97, 98, 135, 136]. However, these approaches to measuring satisfaction are not appropriate when good abandonment is taking place, especially in cases where mobile SERPs are being designed with the explicit goal of satisfying users without them needing to click. It thus becomes necessary to measure user satisfaction in the absence of clicks and recent studies have investigated various click-less approaches for doing this, such as those based on properties of the query [98] and the session [65, 217] and those based on gaze and viewport tracking [159].

We take a different approach and hypothesize that a user's gestures provide signals for detecting user satisfaction. Specifically, we focus on mobile search where gestures are prevalent and seek to answer the following main research question:

> *In the absence of clicks, what is the relationship between a user's gestures and satisfaction and can we use gestures to detect satisfaction and good abandonment?*

In this study, we use the term *gestures* to refer to users' click-less interactions with their mobile devices, such as touch gestures, swipe gestures and reading actions. In addressing this main research question, we focus on three sub-questions:

> **RQ 6.1:** *Do user's gestures provide signals that can be used to detect satisfaction and good abandonment in mobile search?*

**RQ 6.2:** *Which user gestures provide the strongest signals for satisfaction and good abandonment?*

**RQ3 6.3:** *What SERP elements are the sources of good abandonment in mobile search?*

To our knowledge, this is the first work to consider the use of gestures to predict user satisfaction in mobile search and to use it to differentiate between good and bad abandonment. Furthermore, to our knowledge, this is also the first work to measure the relationship between user gestures and good abandonment in mobile search.

In summary, we make the following contributions:

- We construct gesture features for measuring user satisfaction in mobile search.
- We build a classifier that can automatically differentiate between good and bad abandonment and that performs significantly better than several baselines.
- We measure the correlation between user gestures and satisfaction.
- We identify the SERP elements that lead to good abandonment in mobile search.

The remainder of this chapter is organized as follows. Section 6.2 discusses related work and Section 6.3 presents the problem we address. Section 6.4 describes the data we collected for this study and Section 6.5 describes the gesture features that we developed to detect satisfaction and good abandonment. Section 6.6 presents an analysis of the sources of good abandonment and satisfaction in our datasets and Section 6.7 presents the results of experiments for measuring good abandonment. Lastly, conclusions and plans for future work are discussed in Section 6.8.

## 6.2 Background and Related Work

In this section, we will discuss related work relevant to the research described in this chapter, covering three broad strands of research: we discuss satisfaction in search in Section 6.2.1, which supplements our discussion in Section 4.2.2 and 5.2.3; detecting good abandonment in Section 6.2.2; and user gestures in Section 6.2.3.

### 6.2.1 User Satisfaction in Search

Satisfaction is a subjective measure of a user's search experience and has been referred to as the extent to which a user's goal or desire is fulfilled [131]. For instance, satisfaction may be influenced by the relevance of results, time taken to find results, effort spent, or even by the query itself [134]. Thus, satisfaction

is different from traditional relevance measures in information retrieval, such as Precision, MAP and NDCG, which are based on the relevance of results and not on the overall user experience. However, similar to the case for relevance metrics, such as NDCG, satisfaction can also be fine-grained [122] and personalized [96] and it has been shown that search success does not always lead to satisfaction [87].

Several methods for measuring and predicting user satisfaction have been proposed. For instance, it has previously been shown that clicks followed by long dwell times are correlated with satisfaction [77]. Hassan et al. [98] propose to use query reformulation as an indicator of search success and thus satisfaction and show how an approach based on query features outperforms an approach based on click features, with the best performance being achieved by a combination of the two. Like our proposed work, this work does not consider clicks; however, it differs from ours since we consider gestures rather than query reformulation. Furthermore, we focus on good abandonment rather than general satisfaction.

In [97], the search process is modeled as a sequence of actions including clicks and queries and two Markov models are built to characterize successful and unsuccessful search sequences. In [95], a sequence of actions is also considered, but a semi-supervised approach is shown to be useful for improving performance when classifying Web search success.

Kim et al. [135] consider three measures of dwell time and evaluate their use in detecting search satisfaction. In [136] it is shown that the SAT and DSAT dwell times for a page depend on the complexity and topic of a page. To address this issue, the authors propose query-click complexities in modeling dwell times on landing pages. Since we only consider abandoned queries in our study, landing page dwell times do not exist; however, we do consider a similar feature based on visibility and reading times for various elements in a SERP.

### 6.2.2 Good Abandonment

Diriye et al. [65] investigate the rationale for abandonment in search. In a survey involving 186 participants, it was found that satisfaction was responsible for 32% of abandonment. They also studied 39,606 queries submitted to a search engine of which about 22% were abandoned and, for half of the abandoned queries, rationale for abandonment were collected via a popup window. For the cases where feedback was provided, it was found that satisfaction was responsible for 38% of abandonment.

In [219] it was found that 27% of searches were performed with the predetermined goal of having the search satisfied by the SERP and that 75% of searchers were satisfied this way. In [166] it was found that, for queries that could potentially lead to good abandonment, 56% were clearly or possibly satisfied by the SERP on the desktop, and 70% on mobile. The authors hypothesized that one of the reasons for the higher potential abandonment rates on mobile is because users may formulate queries in such a way so as to increase the likelihood of them being answered on the SERP due to a clumsy experience in retrieving webpages for display on mobile. In [51], the effect that answers have on users'

interactions with a SERP is studied and it is observed that the presence of answers *cannabilizes* clicks by reducing interaction with the SERP. A similar finding was presented in [54] where it was found that high quality SERPs decrease click-through rates and increase abandonment. For this reason, we consider features that incorporate non-click interactions with answers, such as element visibility duration and attributed reading time (see Section 6.5.1).

In [217], context is considered in predicting good abandonment. Query-level features, such as query length and reformulation, SERP features that consider clicks in neighboring queries and the presence of answers on a SERP, and session features are used to identify good abandonment. In [55], topical, linguistic features are used to detect potential good abandonment and achieved F-scores of 0.38, 0.55 and 0.71 for maybe, good and bad abandonment, respectively. Our work differs from these approaches in that we use non-click gesture features for detecting good abandonment.

### 6.2.3 Gestures for Relevance and Satisfaction

User gestures have been used in various ways to detect success and satisfaction in search. One of the common approaches is to use scroll and mouse movement behaviors in satisfaction prediction [49, 84, 85, 171]. In [85] post-click behavior, such as scrolls and cursor movement, is used to estimate document relevance for landing pages. In [88] similar features are used to predict session success. Our work differs from this work in that we do not attempt to detect post-click satisfaction, but instead predict satisfaction in the absence of a click. Furthermore, scrolls and cursor movements do not exist in mobile search; however, the swipe interaction performs a similar function and we use swipe interactions as signals for detecting good abandonment.

The two studies most similar to ours evaluate the use of user interaction on mobile phones for detecting search result relevance [90] and use eye- and viewport-tracking to measure user attention and satisfaction [159]. User interactions on mobile phones, such as swipes, dwell times on landing pages and zooms are used in [90] to predict Web search result relevance. While our study uses similar gesture features to [90], our study differs from this since, instead of predicting relevance of landing pages, we differentiate between good and bad abandonment. Furthermore, landing page interactions are used in [90], whereas we use gestures on the SERP itself and do not take visited pages into consideration. Similar features were combined with server-side features such as click-through rate in [87] to predict search success. Once again, our approach differs from this work in that we attempt to predict good abandonment. In [159] viewport- and eye-tracking were used to measure user attention and satisfaction. The authors establish the correlation between gaze time and viewport time and also studied the effect of having relevant/irrelevant answers on the user behavior and the correlation between individual signals and relevance. The authors focus on SERPs containing answer-like results since clicks on these answers do not occur frequently. Through a user study, it was shown that users are more satisfied when answers or knowledge graph information is present in the SERP. Our work differs in that, instead

of only focusing on answers, we consider multiple sources of satisfaction and good abandonment in mobile search; we also consider a large number of gesture-based features beyond gaze and viewport times. Lastly, the authors in [159] suggest building a model to predict satisfaction and good abandonment as a future application; such an application is presented here, through a model for automatically identifying satisfaction and good abandonment using gesture-based features in mobile search.

## 6.3 Problem Description

In this chapter we seek to understand and differentiate between good and bad abandonment in mobile search. We seek to identify the sources of good abandonment, to understand the relationship between user behavior and good abandonment and to identify click-less features that can be used for differentiating between good and bad abandonment.

To address these problems we require a dataset of queries and satisfaction labels, which we collect through a user study and through crowdsourcing. We also require a set of gestures that can be used as signals for measuring satisfaction, which we develop as part of this study. In the following sections, we present the datasets that we created as well as the features we identified for measuring satisfaction.

## 6.4 Data Sets

To collect data to understand good abandonment in mobile search, we conducted a focused user study whereby users completed a set of search tasks and provided satisfaction ratings. The user study setup is discussed in details in Section 4.4. Further, we give a short overview in Section 6.4.1. This led to a dataset of high quality user supplied data that we use for our analysis. However, this dataset is relatively small; thus, we also collected a second dataset via crowdsourcing that we use to validate our findings. Section 6.4.2 describes our crowdsourcing procedure.

### 6.4.1 User Study

We recruited 60 participants from the United States where 75% of them were male and 25% female. The majority (82%) of participants were from a computer science background and the remaining 18% specified their background as either mathematics, electrical engineering or other. English was the first language for 55% of the participants and the mean age was 25.5 ($\pm$5.4) years.

In the user study, 5 information-seeking tasks, which represent atomic information needs [168], were designed in such a way that they may lead to good abandonment. The tasks were not designed to encourage exploration, but rather to allow the user to answer a question. They were:

Table 6.1: SAT Rating Distribution

| SAT Rating | Number of Tasks |
|---|---|
| 1 | 14 |
| 2 | 19 |
| 3 | 47 |
| 4 | 82 |
| 5 | 112 |

1. A conversion between the imperial and metric systems.
2. Determining if it was a good time to phone a friend in another part of the world.
3. Finding the score from a recent game of the user's favorite sports team.
4. Finding the user's favorite celebrity's hair color.
5. Finding the CEO of a company that lost most of its value in the last 10 years.

At the completion of each task users were asked to provide a satisfaction rating on a 5-point scale, specify if they were able to complete the task and the amount of effort required and provide feedback on which element of the SERP they found the information they were looking for and the query that led to them being satisfied.

### Data Description

In the user study, the total number of potential abandonment tasks was 274. A total of 607 queries were submitted for these tasks, with the minimum, maximum, mean and median number of queries per task being 1, 9, 2.2 and 2, respectively. Of the 607 queries, 576 were classified as abandoned queries since they received no clicks.

The SAT distribution (on a scale of 1-5) is shown in Table 6.1. As can be seen from the table, SAT ratings of 4 and 5 make up the majority of the task satisfaction labels. In this study, we follow the approach in previous studies [87, 123] and binarize these values and consider ratings of 4 and 5 as SAT and the remainder of the ratings as DSAT. With this binarization, there are 194 SAT tasks and 80 DSAT tasks.

### Label Attribution

Labels in the user study were collected at the task level. However, good abandonment takes place at a query level. Thus, a way is needed to attribute labels to individual queries. Since users were asked to stop when they found the information they were looking for, the method for doing this is based on the observation that, if a user continues querying then they are likely not satisfied; however, when

a user stops querying then they are either a) giving up the task or, b) satisfied. Based on this observation, individual impressions were labeled as follows: If the task was assigned a DSAT label, then every query for that task was assigned DSAT. If the task was assigned a SAT label, then the final query for the task was assigned the SAT label and every query before it was assigned DSAT. The assumption here is that the queries lead to DSAT until the user meets his information need at which point the query leads to SAT. After filtering queries for which not all features were available, we retained a total of 563 queries of which 461 were abandoned queries.

### 6.4.2 Crowdsourcing

The data collected in the user study is of high quality since users could directly provide information on their satisfaction; however, with only 607 queries, this dataset is relatively small. We thus collected a second set of labeled data via crowdsourcing, which is a common approach to collecting labeled data [240] and that we use to validate our findings. This section describes the collection of that data.

#### Approach

Since our focus is on good abandonment, we randomly sampled abandoned queries from the search logs of a personal digital assistant during one week in June 2015. We filtered the data such that: no adult queries were sampled; all queries originated from within the United States; all queries were input via speech or text; and all queries generated a SERP containing organic Web results and possible answers.

We made use of judges on a commercial crowdsourcing platform. Judges were shown a video explaining the task and how to judge queries with good or bad abandonment, for instance, by considering the query and the SERP and by taking the query context into consideration. Judges needed to pass qualification tasks in order to participate in labeling real data and the crowdsourcing engine had built in spam detection. For each query randomly sampled from the logs, judges were shown: the query, a screenshot of the mobile SERP returned for that query, the previous query in the session and the next query in the session. Judges were asked to provide two judgments: 1) their perception of user-satisfaction on a 5-point scale and 2) if they believed the user was satisfied, which we defined as the user finding the information they were looking for, which type of element on the SERP satisfied the user. Though we asked judges to provide feedback on a 5-point scale, we binarized the labels in the same way as the user study data such that a rating of 1-3 was considered DSAT and 4-5 was considered SAT. We had up to 3 judges provide labels for each query and took the majority vote.

#### Data Description

We gathered a total of 3,895 labeled queries. Among the first two judgments collected for each query, the judges agreed on the label 73% of the time. We

measured inter-rater agreement using Fleiss' Kappa [76], which allows for any number of raters and for different raters rating different items. This makes it an appropriate measure of inter-rater agreement in our study since different judges provided labels for different items. A kappa value of 0 implies that any rater agreement is due to chance, whereas a kappa value of 1 implies perfect agreement. In our data, $\kappa = 0.46$, which, according to Landis and Locke [163], represents moderate agreement. This relatively low $\kappa$ is indicative of a difficult task. After filtering queries for which not all features were available, we retained 1,565 queries for which the judgment was SAT and 1,924 queries for which the judgment was DSAT.

## 6.5 Gestures as Satisfaction Signals

Click signals are not available for measuring satisfaction in abandoned queries. This section describes a set of click-less features that we developed as signals for measuring satisfaction and thus good abandonment.

### 6.5.1 Gesture Features

One of the main contributions of this study is the use of gesture features for detecting good abandonment and satisfaction on mobile devices. Specifically, we focus on gesture features related to the way in which the user interacts with the screen and features based on the elements visible to the user. As noted in [107], capturing touch events is difficult in practice; however, it is possible to infer touch-based interactions based on the mobile viewport, which is the visible region on the device. For instance, if an element is visible in the viewport at some point in time and then no longer visible, one can infer that a gesture must have taken place.

Table 6.2 lists the features used in this study. As previously specified, we use the term *gestures* to refer to touch- and reading-based actions. We also group element visibility features with gesture features since the visibility of an element may imply reading. We separate our features into 6 categories: viewport features (VP); first visible answer features (FA); aggregate answer features (A); aggregate organic search result features (O); focus features (F); and query-session features (QS). We describe these features now.

#### Viewport Features

Viewport features, which are represented by features VP1-VP9 in Table 6.2, capture the user's overall touch gestures with their mobile device. Swipes refer to the gesture whereby the user *swipes* on their device screen to move the content that is visible on the screen. We count the total number of swipes (VP1), the number of up swipes (VP2) and the number of down swipes (VP3). We also count the number of times the user changed swipe direction (VP4), i.e., a down swipe followed by an up swipe or vice versa. We also measure the total distance in pixels swiped on the screen (VP5) and the average distance per swipe (VP6). These

Table 6.2: Description of features used in this study. The last two columns show the correlation with satisfaction (SAT) for both the data gathered in the user study and the data gathered via crowdsourcing. Missing values (-) indicate that the correlation was not statistically significant ($p > 0.05$)

| | **Feature Description** | **User SAT Correlation** | **Crowd SAT Correlation** |
|---|---|---|---|
| VP1 | Total number of swipe actions | -0.08 | -0.14 |
| VP2 | Number of up swipe actions | - | -0.04 |
| VP3 | Number of down swipe actions | -0.08 | -0.15 |
| VP4 | Number of swipe direction changes | - | -0.09 |
| VP5 | The total distance swiped in pixels | -0.10 | -0.14 |
| VP6 | The average swipe distance | -0.10 | - |
| VP7 | The dwell time on the SERP | - | - |
| VP8 | The mean dwell time on SERP before or after each swipe | - | - |
| VP9 | Total swipe distance divided by time spent on the SERP | -0.11 | -0.11 |
| FA1 | Attributed reading time (RT) for the first visible answer | - | 0.04 |
| FA2 | Attributed reading time per pixel (RTP) of the first answer | 0.10 | 0.08 |
| FA3 | The duration for which the first answer was shown | - | 0.06 |
| FA4 | The fraction of visible pixels belonging to the first answer | - | 0.15 |
| A1-A4 | Max, min, mean and SD attributed RT for answers | -/-/-/- | 0.04/-/-/0.04 |
| A5-A8 | Max, min, mean and SD attributed RTP for answers | 0.11/0.11/0.11/- | 0.08/0.06/0.07/0.04 |
| A9-A12 | Max, min, mean and SD shown duration for answers | -/-/-/- | 0.04/0.05/0.05/- |
| A13-A16 | Max, min, mean and SD shown fraction for answers | -/-/-/- | 0.15/0.11/0.14/0.10 |
| O1-O4 | Max, min, mean and SD RT for organic results | -/-/-/- | -0.15/-/-0.09/-0.12 |
| O5-O8 | Max, min, mean and SD RTP for organic results | -/0.10/-/- | -0.13/-/-0.06/-0.12 |
| O9-O12 | Max, min, mean and SD shown duration for organic results | -/-/-/- | -/-/-/- |
| O13-O16 | Max, min, mean and SD shown fraction for organic results | -0.20/-0.19/-0.29/0.10 | -0.20/-0.07/-0.22/-0.05 |
| F1 | Time to focus on an answer | - | -0.05 |
| F2 | Time to focus on an organic search result | - | - |
| QS1 | Session duration | - | - |
| QS2 | Number of queries in session | -0.16 | - |
| QS3 | Index of query within session | -0.24 | - |
| QS4 | Query length (number of words) | -0.17 | -0.26 |
| QS5 | Is this query a reformulation? | -0.11 | -0.10 |
| QS6 | Was this query reformulated? | -0.35 | -0.15 |
| QS7 | Time to next query | 0.16 | -0.04 |
| QS8 | Click count | - | - |
| QS9 | Number of clicks with dwell time > 30 seconds | - | - |
| QS10 | Number of clicks followed by a back-click within 30 seconds | - | - |

features capture the number of SERP features seen by the user. We capture the total time spent on the SERP (VP7) and also the average amount of time between swipes (VP8), which captures how long the user spent looking at the screen after it changed. Lastly, we capture the swipe speed (VP9) as it is has been shown that slow swipes are associated with reading and fast swipes are associated with skimming [90].

### First Answer Features

One of our hypotheses in conducting this study was that the highest ranked visible answer on a SERP, by nature of being highly ranked, has the highest likelihood of satisfying the user. Thus, we capture a set of features that relate to the first visible answer on a SERP. We estimate the attributed reading time for the first visible answer on the SERP (FA1). We calculate attributed reading time for answer $e$, $ART_e$ as:

$$ART_e = \sum_{v \in V} t_v \times \frac{AA_{e,v}}{VA_v}, \tag{6.1}$$

where $V$ is the set of viewport instances, $t_v$ is the duration of time for which viewport $v$ was visible and $AA_{e,v}$ and $VA_v$ are the visible areas of answer $e$ and viewport, respectively, in the viewport $v$. We also attribute a reading time to each pixel belonging to the first answer (FA2). We calculate the attributed reading time per pixel for an answer $e$, $RTP_e$ as:

$$RTP_e = \frac{1}{AA_{e,O}} ART_e, \tag{6.2}$$

where $AA_{e,O}$ is the pixel area of the answer $e$ that was ever observable by the user across all viewports corresponding to the impression.

We calculate the total duration for which the first answer is (even partially) shown (FA3), which differs from attributed reading time since it is not scaled according to the visible area of the answer. Lastly, we calculate the fraction of visible pixels belonging to the first answer (FA4) as $\frac{AA_{e,O}}{AA_e}$, where $AA_e$ is the physical pixel area of the underlying answer, observed or not.

### Aggregate Answer Features

Features FA1-FA4 related specifically to the first visible answer on a mobile SERP. Features A1-A16 are similar in this regard, except that they aggregate and provide descriptive statistics based on the set of answers visible on a SERP. Specifically, we calculate the min, max, mean and standard deviation of the following features for the set of answers: attributed reading time (A1-A4); attributed reading time per pixel (A5-A8); total duration shown (A9-A12); and fraction of visible pixels (A13-A16).

#### Aggregate Organic Result Features

We also aggregate the same set of features for organic search results by calculating the min, max, mean and standard deviation of the following for visible organic search results: attributed reading time (O1-O4); attributed reading time per pixel (O5-O8); total duration shown (O9-O12); and fraction of visible pixels (O13-O16).

#### Time to Focus Features

We define two *time to focus* features. These features capture how long it takes a user to focus on a page element where we define focus as occurring when an element is visible for some minimum amount of time, which we set to 5 seconds. When an element has been visible for 5 seconds, we set the time to focus as the timestamp at which the element first became visible. We calculate the time to focus on an answer (F1) and an organic search result (F2).

### 6.5.2 Query and Session Features

While the main contribution of this work is in the gesture features, it has previously been shown that other user behavior also provides strong signals for satisfaction [96, 97]. Thus, we also use a set of features based on the query and the user behavior within the session. these features are shown by features QS1-QS10 in Table 6.2 and are self-explanatory.

### 6.5.3 Endogenous and Exogenous Features

The features used in this study were designed to be *exogenous*, meaning that the system does not have direct control over them but that instead the features are based on user input, such as swipe actions and dwell times. This is in contrast to *endogenous* features that the system can directly influence. For instance, the number of answers shown on a SERP or the presence of a certain answer type, e. g. weather, are examples of features that are likely to be endogenous. While endogenous features are useful for measuring satisfaction, they present a challenge for search engine evaluation since a system change can be unintentionally optimized for these features. As an example, if the presence of a weather answer is an indicator of satisfaction, then an answer ranker may learn to always rank weather answers highly thereby *gaming* the metric. Though we found endogenous features to be very useful for detecting good abandonment, for the reasons described above we choose to only use features that are mostly exogenous in this study. It is important to note, though, that the classification of endogenous and exogenous features is not absolute, but rather falls along a spectrum depending on the search engine and metric, and that the classification will differ depending on the circumstances.

## 6.6 Good Abandonment, Interaction and User Satisfaction on Mobile Devices

In this section, we present the reasons for good abandonment and show which user gestures are correlated with good abandonment in Section 6.6.1. We also investigate the relationship between satisfaction and other feedback collected from users in Section 6.6.2 and 6.6.3.

### 6.6.1 Causes of Good Abandonment

The main contribution of this research is an investigation into the use of gesture features to detect good abandonment. One of the first stages in doing this is understanding the causes of good abandonment. This allows us to consider the contents of a SERP when trying to determine if a query was abandoned because the user is satisfied without the need to click. Thus, in the user study, we asked users to provide feedback on the source of satisfaction. The users were asked to select from among the following:

- **Answer.** An answer on the SERP.
- **Search Result Snippet.** The text appearing below a search result.
- **Image.** An image displayed on the SERP.
- **Website.** If the user visited a Website to satisfy his information need.
- **Other.** An element on the SERP that does not belong to one of the above categories.

As can be seen from Figure 6.2, the majority of user satisfaction (56%) was due to answers on the SERP. However, an important observation is that good abandonment can be due to other sources on the SERP. For instance, images made up for 7% of satisfaction and snippets made up 11% of satisfaction. Since users were allowed to click on search results, websites were responsible for 25% of satisfaction, which is less than half of the number of times users were satisfied by answers. This analysis provides an answer to **RQ3 6.3:** *What SERP elements are the sources of good abandonment in mobile search?* It confirms our hypothesis that there are many sources of satisfaction on a SERP.

Figure 6.3 shows the user satisfaction associated with each of the sources of satisfaction. The mean is represented by the dot and the median by the horizontal line. As can be seen from the figure, the satisfaction ratings are highest for the answers on the SERP, and the means for images and snippets are relatively close to that for answers. The mean for websites is the lowest since users have to visit websites without knowing if it will satisfy them.

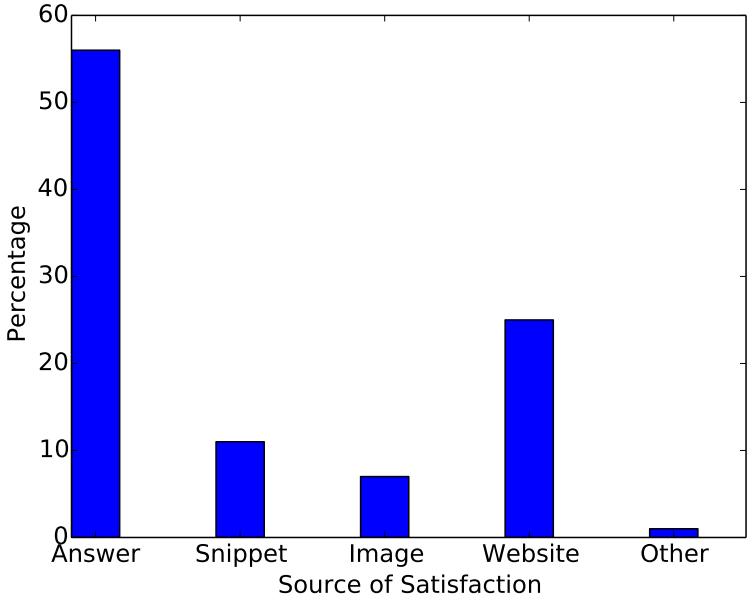



Figure 6.2: A comparison of the counts of the sources of satisfaction from the user study.

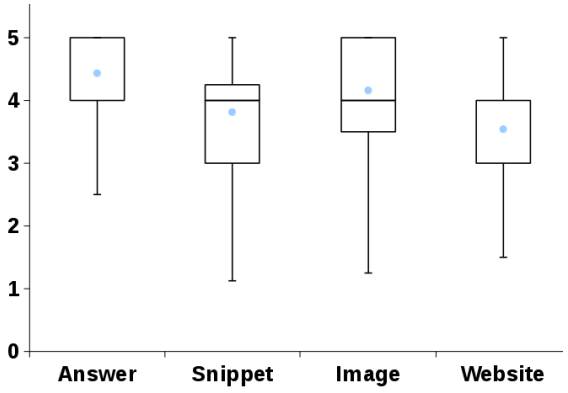



Figure 6.3: Satisfaction associated with each source of information.

### 6.6.2 Gesture Features and User Satisfaction

To better understand the relationship between gestures and good abandonment and satisfaction, we calculate the Pearson correlation between the satisfaction label and each feature. The statistically significant correlations ($p < 0.05$) for the user study data and crowdsourced data are shown in the two last columns of Table 6.2 where a missing value (-) indicates that the correlation was not significant ($p > 0.05$).

As can be seen from Table 6.2 there are several features that are significantly correlated with SAT. For instance, features from the crowdsourced data related to swipes such as the total number of swipes, the number of down swipes and the distance swiped are all negatively correlated with satisfaction. We note that one limitation with this observation is that judges were only presented with screenshots of the mobile SERP and thus were unable to swipe to see if there was additional information on the SERP not shown in the screenshot that may have satisfied the user. That being said, a similar trend is observed for the user study data where users were able to swipe. For instance, for the user study both the total number of swipes and the number of down swipes are negatively correlated with satisfaction. Furthermore, a similar finding was presented in [159] where it was shown that scrolling is negatively correlated with user satisfaction. The fact that the swipe action is negatively correlated with satisfaction suggests that the more time that users spend physically touching and moving the viewport on a mobile device, the less likely they are to be satisfied. One reason that this may be the case is that, as shown in Figure 6.2, a lot of good abandonment is due to answers and, when an answer is present on the viewport there may be less reason for the user to physically interact with the SERP.

Features related to the reading and visibility of answers (features FA1-F4; A1-A16), when statistically significant, are all positively correlated with satisfaction. This implies that the longer users spend viewing answers, the more likely they are to be satisfied. This is interesting when contrasted with feature VP7, which is the total time spent on the SERP, and which is not statistically significant. The data suggests that the time spent on a SERP is not a strong signal for satisfaction but that the time spent viewing answers is.

The opposite effect is observed when considering the correlation between satisfaction and the time spent reading and viewing organic search results (features O1-O16). When significant, increased interaction with organic search results is negatively correlated with satisfaction. Increased interaction with organic search results may imply that users are spending more time on the SERP unsuccessfully looking for information to satisfy their information needs.

The analysis above provides an answer to **RQ 6.2:** *Which user gestures provide the strongest signals for satisfaction and good abandonment?* Features related to swipe actions and interaction with organic search results provide indications of bad abandonment. On the other hand, extended reading-based interactions with answers on a SERP are signals that suggest good abandonment.

Table 6.2 also shows that the correlation between satisfaction and features based on the query and session (QS1-QS10). Our finding confirms existing find-

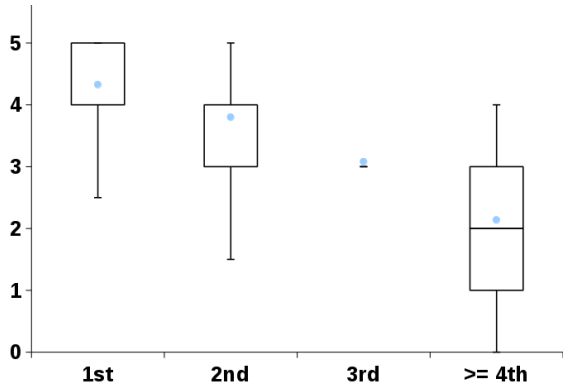



Figure 6.4: The relationship between query number and satisfaction.

ings in the literature, such as the fact that query length and reformulation are negatively correlated with satisfaction [98, 217] and, as in [217], we find in our user study data that the time to a next query is positively correlated with satisfaction though we observe the opposite effect in our crowdsourced data.

### 6.6.3 User Feedback and Good Abandonment

In addition to asking users how satisfied they were and where they found the information they were looking for, we also asked them: (a) if they were able to complete the task, (b) how much effort they put into the task and (c) which query led to finding the answers, with them being able to specify first, second, third or fourth or later. We find strong significant negative correlation of -0.65 between satisfaction and effort, and a negative correlation of -0.08 between completion and effort, indicating that less effort leads to more satisfaction and higher completion rates.

Figure 6.4 shows the relationship between satisfaction and the number of queries submitted by the user. As can be seen from the figure, there is a negative relationship between the number of queries required to satisfy the user's information need and their level of satisfaction. This finding makes sense for information seeking tasks, such as those used in this user study; however, we suspect that for exploratory tasks this finding may not always hold; we leave this to future work.

## 6.7 Classifying Abandoned Queries

The previous section presented an analysis of the reasons for good abandonment and which behaviors are correlated with satisfaction. In this section, we present our approach to differentiating between good and bad abandonment in Section 6.7.1. Our baselines are discussed in Section 6.7.2. The proposed models to detect good abandonment is presented in Section 6.7.3. Finally, results are

presented in Section 6.7.4 and discussed in Section 6.7.5.

### 6.7.1 Approach

We formulate a supervised classification problem where, given an abandoned query, the goal is to classify the query as being due to good abandonment or not. We use a random forest classifier, which is an ensemble classifier made up of a set of decision trees [39]. Each tree is built with a bootstrap sample from the dataset and splitting in the decision tree is based on a random subset of the features rather than the full feature set [74]. In this study, the number of trees in the ensemble is set to 300 since this was empirically found to perform well and the number of features randomly selected is equal to $\sqrt{n\_features}$. At each level in the decision trees, variables are selected for splitting with the Gini index. The Gini index is defined as follows:

$$I_G(i) = \sum_{j=1}^{K} p_j(1 - p_j) = 1 - \sum_{j=1}^{K} p_j^2, \tag{6.3}$$

where $K$ is the number of classes and $p_j$ is the proportion of instances belonging to class $j$ in node $i$. If a node $i$ is pure (only contains one type of class), then $I_G(i) = 0$. The Gini index is used in decision tree learning for selecting the variable to split on at each node, with the split that leads to the largest reduction in the Gini index being selected.

We use 10-fold cross validation and use grid search to optimize for the number of leaves, tree depth and number of leaves required to split for each training fold. During training, we downsample the majority class so that our class representation is even; however, we leave the class distribution unchanged in the testing data. Since we do random downsampling of training data, we repeat each experiment 100 times and report the average. For our experiments, we make use of 3 baselines and propose 2 new models.

### 6.7.2 Baselines

#### Click and Dwell with no Reformulation

This baseline is based on the common approach in the literature as labeling satisfaction as occurring if a user clicks on a search result and then spends a minimum of $t$ seconds on a page and does not follow the query up with a reformulation. Spending a minimum amount of time on a webpage is known as a long dwell click and has been shown to be correlated with satisfaction [77]. In this study, we set $t = 30$ seconds. Naturally, this baseline does not make much sense for the detection of good abandonment since, by definition, abandoned queries do not have any clicks. Nonetheless, it is useful to use this baseline for comparison so as to show why click-based metrics are not appropriate.

#### Optimistic Abandonment

Baseline 2 is an optimistic one whereby, if there is no click and no reformulation, then it is assumed that the abandonment is good. We refer to this baseline as optimistic since it optimistically assumes that all abandonment without reformulation is good. For queries that receive clicks, the same approach as in Baseline 1 is used to measure satisfaction.

#### Query-Session Model

Baseline 3 makes use of features from the literature for detecting satisfaction and good abandonment. Specifically, it is a supervised classifier based on features QS1-QS10 in Table 6.2 that represent the query and the session.

### 6.7.3 Proposed Models

#### Gesture Model

This is a supervised classifier based only on the interaction features in Table 6.2, which is all except features QS1-QS10. The purpose of this model is to only consider the users physical behavior and gestures with the screen and investigate their usefulness in detecting good abandonment.

#### Gesture + Query-Session Model

This is a supervised classifier that combines the interaction-features model and the query-session model.

### 6.7.4 Results

We present three sets of results. First, we present results using only abandoned queries from the user study. Secondly, since the user study dataset is relatively small, to validate our approach we repeat the experiment using the crowdsourced data. Lastly, even though the focus of this study is on good abandonment, it is also useful to investigate the use of click-less interaction features for detecting satisfaction in general. Thus, we also present satisfaction detection results on all data from the user study, which includes both abandoned and non-abandoned queries. For each experiment we report the overall accuracy as well as the precision (P), recall (R) and $F_1$ score for SAT and DSAT separately.

#### Abandoned User Study Queries

Table 6.3 shows the performance on abandoned queries from the user study. As can be seen from the table, the highest accuracy of 75% is achieved by the model that combines gesture features with query-session features and is significantly better ($p < 0.01$) than the accuracy achieved by all other models. The approach based on query and session features from the literature achieves an accuracy of 73% and the gesture features alone achieve an accuracy of 70%. While the

Table 6.3: Performance of various classifiers on only abandoned user study data

| Classifier | Acc | SAT P | DSAT P | SAT R | DSAT R | SAT F1 | DSAT F1 |
|---|---|---|---|---|---|---|---|
| Click & Dwell | 0.68 | 0.00 | 0.68 | 0.00 | **1.00** | 0.00 | **0.88** |
| Optimistic | 0.61 | 0.45 | **0.93** | **0.93** | 0.46 | 0.61 | 0.62 |
| Query-Session (QS) | 0.73 | 0.56 | 0.87 | 0.77 | 0.71 | 0.65 | 0.78 |
| Gesture | 0.70 | 0.53 | 0.84 | 0.70 | 0.70 | 0.60 | 0.76 |
| Gesture + QS | **0.75** | **0.59** | 0.88 | 0.78 | 0.74 | **0.67** | 0.80 |

Table 6.4: Performance of various classifiers on crowdsourced data

| Classifier | Acc | SAT P | DSAT P | SAT R | DSAT R | SAT F1 | DSAT F1 |
|---|---|---|---|---|---|---|---|
| Click & Dwell | 0.55 | 0.00 | 0.55 | 0.00 | **1.00** | 0.00 | **0.71** |
| Optimistic | 0.53 | 0.49 | 0.71 | **0.88** | 0.25 | 0.63 | 0.37 |
| Query-Session (QS) | 0.64 | 0.59 | 0.69 | 0.66 | 0.63 | 0.62 | 0.66 |
| Gesture | 0.64 | 0.59 | 0.69 | 0.65 | 0.62 | 0.62 | 0.65 |
| Gesture + QS | **0.68** | **0.63** | **0.73** | 0.69 | 0.67 | **0.66** | 0.70 |

Table 6.5: Performance of various classifiers on all user study data

| Classifier | Acc | SAT P | DSAT P | SAT R | DSAT R | SAT F1 | DSAT F1 |
|---|---|---|---|---|---|---|---|
| Click & Dwell | 0.66 | 0.27 | 0.68 | 0.67 | **0.94** | 0.10 | **0.79** |
| Optimistic | 0.61 | 0.44 | **0.87** | **0.84** | 0.50 | 0.58 | 0.63 |
| Query-Session (QS) | 0.69 | 0.52 | 0.84 | 0.72 | 0.68 | 0.60 | 0.75 |
| Gesture | 0.66 | 0.48 | 0.80 | 0.64 | 0.67 | 0.55 | 0.73 |
| Gesture + QS | **0.72** | **0.55** | 0.85 | 0.73 | 0.71 | **0.62** | **0**.77 |

accuracy achieved by the gesture features is not as high as that achieved by the query-session features, it is still very interesting to note that, using only gesture features, it is possible to differentiate between good and bad abandonment with 70% accuracy and that this approach is significantly better ($p < 0.01$) than the other two baselines.

Table 6.3 also shows precision, recall and F1 scores for SAT and DSAT. As would be expected, the first baseline based on click and dwell performs very badly on SAT since there are no clicks. Thus, while it results in the highest F1 score for DSAT, the F1 score for SAT is 0. The optimistic baseline overestimates SAT and thus has low SAT precision but high SAT recall. However, this comes at the expense of having the lowest DSAT recall and lowest accuracy overall.

The model that combines query-session and gesture features achieves the second highest $F_1$ score for DSAT and the highest $F_1$ score for SAT. In fact, the model performs either best or second best for every metric and the best overall if one considers the accuracy or the $F_1$ scores.

### Crowdsourced Data

To validate our model, we also consider differentiating between good and bad abandonment in the data gathered via crowdsourcing. Table 6.4 shows the performance. As can be seen from the table, as was the case with the user study data, the best accuracy of 68% is achieved by combining gesture and query-session features and is significantly better than all other methods ($p < 0.01$). Interestingly, for this data, the gesture features perform as well as the query-session features, with both methods achieving accuracies of 64% and both outperforming the other baselines. Overall, the query-session model and the gesture models achieve similar performance across all metrics.

As was the case with the user study data, the click & dwell baseline is unable to detect SAT since all of the queries are abandoned and have no clicks. Similarly, the optimistic baseline performs relatively poor when it comes to its precision in detecting SAT since it overestimates good abandonment in the data; however, for this reason it achieves the highest SAT recall but the lowest DSAT recall.

The combination of query-session and gesture features achieves the highest precision for both SAT and DSAT as well as the best recall and $F_1$ score if one averages the values for SAT and DSAT.

### All User Study Queries

To show the appropriateness of interaction features for detecting other types of satisfaction in addition to good abandonment, we also run a classification experiment on all data from the user study, which includes some queries that had clicks. Table 6.5 shows the performance on this data. As can be seen from the table, the highest accuracy when not including gesture-interaction features is 69% and is achieved by making use of the third baseline, which uses query-session features. The other baselines achieve accuracies of 66% and 61%, respectively. When only interaction features are considered, the accuracy is 66%, which is equal to the

accuracy achieved by the click and dwell baseline, but less than the query-session features. However, when gesture features are combined with query-session features, the accuracy increases to 72%, which is statistically significantly better ($p < 0.01$) than all the other approaches. This combined model also achieves the highest SAT precision and F1 score, and performs second best for all other metrics.

While this chapter has focused on detecting good abandonment, this experiment has shown that the gesture features are useful for detecting satisfaction in general. We expect this to be an interesting area for future research.

### 6.7.5 Discussion and Implications

We have presented various experiments for differentiating between good and bad abandonment. Our main finding is that gesture features are useful for accomplishing this goal, often achieving the same or very similar performance to an approach based on query and session features. Overall though, the best performance comes from combining these gesture features with query-session features. The reason for this is that gesture features provide us with signals that we may not be able to get from the query or session. For instance, reformulation is usually considered a strong signal for DSAT; however, the absence of reformulation does not necessarily imply SAT as was the assumption in our second baseline, which was an optimistic classifier. Instead, our findings suggest that combining signals, such as the fact that the user did not reformulate, with information on how the user interacted with the screen is more powerful.

While this study has focused on detecting good abandonment, our experiment considering all of the user study data showed that interaction features were also useful for detecting satisfaction when clicks existed and outperformed the baseline based on a click followed by a long dwell. We believe that it will be useful to consider gesture features for general satisfaction prediction and leave this for future work.

The implications of our experiments is two-fold. Firstly, it is important to develop click-less models that are able to capture satisfaction due to good abandonment. Secondly, we have shown that, while session and query features are useful for differentiating between good and bad abandonment, the inclusion of gesture features can successfully be used to improve good-abandonment detection.

As discussed in Section 6.5.3, in this study we focused on using exogenous features, which are more difficult for the ranker to optimize for. This is in contrast to endogenous features, such as the presence of certain answer types or the number of search results displayed on the page. However, to estimate an upper bound on an accuracy that may be feasible to achieve with the collected data, we also conducted an experiment where we additionally considered a set of endogenous features. Specifically, we include the following endogenous features: the number of answers and organic results on the SERP; the number of answers and organic results that came into view; the fraction of the number of answers and organic results that were visible; binary features indicating the presence of dif-

ferent answer types on the SERP, such as weather, currency, etc. Using these endogenous features, we achieve an accuracy of 78% on the user study data and an accuracy of 70% on the crowdsourced data. Both of these models demonstrate improvements over models where only exogenous features are used; however, as previously discussed, it is often undesirable to use exogenous features since a ranker may unintentionally optimize for them.

## 6.8 Conclusions

This chapter proposed the use of gesture features for differentiating between good and bad abandonment in mobile search. We sought to answer three research questions, the findings of which we summarize below.

**RQ 6.1:** *Do user's gestures provide signals that can be used to detect satisfaction and good abandonment in mobile search?*

By formulating a supervised classification experiment, we showed how user gesture features perform significantly better than query and session features as well as other click-based and optimistic baselines. We show this on a high quality dataset collected through a user study and verify the results on a crowdsourced dataset.

**RQ 6.2:** *Which user gestures provide the strongest signals for satisfaction and good abandonment?*

Through a correlation analysis, we showed how time spent interacting with answers on a SERP are positively correlated with satisfaction and good abandonment. By contrast, swipe interactions and time spent interacting with organic search results are negatively correlated with satisfaction.

**RQ3 6.3:** *What SERP elements are the sources of good abandonment in mobile search?*

By analyzing data collected through our user study, we showed how good abandonment can be driven by many elements on a SERP, such as answers, snippets and images and conclude that good abandonment is due to many factors.

An interesting problem for future work would be to attribute the good abandonment to a specific entity on the screen. For instance, one might consider the attributed reading time for each element and use this information to infer which element led to good abandonment. Furthermore, it will be interesting to analyze how users' behavior differs in the presence of different entity types on the screen. This work has been performed exclusively on mobile devices, but many of the conclusions are likely transferable to tablet or desktop search; we leave this for future investigations.

# Part III

# Predicting User Satisfaction on the SERP-level

# Part III
# Predicting User Satisfaction on the SERP-level

In the third and final part of this dissertation, we study behavioral dynamics—changes in aggregated user behavioral features over time. We look for indicators of a drop in user satisfaction, due to a SERP trained on historical data becoming outdated as a result of changing query intents over time or due to external context (e. g. news events). Specifically, Part III deals with the following research question:

> **RQ 4**: *How to define and to detect changes in user satisfaction with retrieved search results?*

Chapter 7 investigates how the query reformulation signal can be used to detect drifts in user satisfaction with the current SERP for the query, using concept drift detection approach from online supervised learning, and study its effectiveness on search log data from Microsoft Bing. Chapter 7 is based on [145].

Chapter 8 extends this method focusing on failed SERPs and taking into account more signs of user frustration (or lack of search satisfaction) such as: a rate of search abandonment, a dramatic change in query volume, or a lowering in average click positions, and conducts a large-scale evaluation with one year of search log data from Yandex. Chapter 8 is based on [148].

# 7 Query Reformulations

Informational needs behind queries, that people issue to search engines, are inherently sensitive to external factors such as breaking news, new models of devices, or seasonal changes as *'black Friday'*. Mostly these changes happen suddenly and it is natural to suppose that they may cause a shift in user satisfaction with presented old search results and push users to reformulate their queries. For instance, if users issued the query *'CIKM conference'* in 2013 they were satisfied with results referring to the page `cikm2013.org` and this page gets a majority of clicks. However, the conference site has been changed and the same query issued in 2014 should be linked to the different page `cikm2014.fudan.edu.cn`. If the link to the fresh page is not among the retrieved results then users will reformulate the query to find desired information.

In this chapter, we examine how to detect changes in user satisfaction if some events affect user information goals but search results remained the same. We formulate a problem using concept drift detection techniques. The proposed method works in an unsupervised manner, we do not rely on any labelling. We report results of a large scale evaluation over real user interactions, that are collected by the commercial search engine within six months. The final datasets consist of more than sixty million log entries. The results of our experiments demonstrate that by using our method we can accurately detect changes in user behavior. The detected drifts can be used to enhance query auto-completion, user satisfaction metrics, and recency ranking.

## 7.1 Introduction

Millions of users interact with search engines daily to obtain *fresh* information quickly while minimizing their effort. Users issue a query $Q$ and a search engine returns search result page ($SERP$) that is a ranked list of URLs: $SERP = (url_1, \ldots, url_i, \ldots, url_n)$.

The order of URLs in $SERP$ is optimized to fit a history of user interactions with a pair $\langle Q, SERP \rangle$ [12]. However, events from the outside world and time

can affect user behavior on the Web [157, 185]. To illustrate this drift in the user information goals let us consider the following examples:

- The last Olympics games occur in 2014, so users are not interested in the previous 2012 Olympics anymore (if users do not specify which games they interested in: winter or summer then search engines returns information about last available one). Therefore, if users issue the query *'Olympic games'* in 2014 they need to find a page of the latest event. If the desired link is not among the retrieved results then user satisfaction with the served $SERP$ decreases.
- After Microsoft releases a new *'Windows phone 8'* users are not satisfied if pages of previous models are in the top of $SERP$.
- After a cartoon 'Despicable Me 2' is released the audience pays less attention to its previous release.

User satisfaction with a pair $\langle Q, SERP \rangle$ can decrease dramatically if user information needs change due to some event or decay/change of interest over time. In this chapter, we answer the question:

> *How can we detect a drift in user satisfaction with the pair $\langle Q, SERP \rangle$ using users' interactions on the $SERP$?*

We break up our main research problem into three different parts. Our first concrete research question is:

> **RQ 7.1:** *What behavioral signal can be used to infer changes in user satisfaction?*

When users struggle to find an answer for $Q$ they run a follow-up query $Q'$ that is an expansion of $Q$. *Query reformulation* is the act of submitting a next query $Q'$ to modify a previous $SERP$ for a query $Q$ in the hope of retrieving better results [98]. Such a query reformulation is a strong indication of user dissatisfaction [9]. We call this the *reformulation signal*. Our hypothesis is that a decrease in user satisfaction with $\langle Q, SERP \rangle$ correlates nicely with the reformulation signal. In other words, the probability of reformulating $Q$ will grow dramatically.

Let us consider the probability of reformulating a query *'flawless'* during year 2013. A histogram of this probability is shown in Figure 1. We can clearly see that a drift happened in December. When users ran this query before October 2013 they most probably were looking for a movie, called *'flawless'*. However, the singer Beyonce released her new soundtrack also called *'flawless'* in November 2013. Hence, this event affected dramatically the meaning of this query. As a result, if the desired song was missing in $SERP$ a majority of users reformulated the query by expanding it with the term *'Beyonce'*.

Our second concrete research question is:

> **RQ 7.2:** *How can we detect changes in user satisfaction using reformulation signal?*

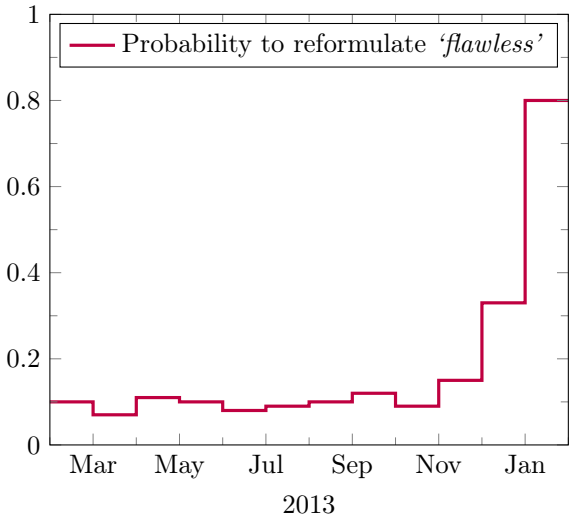



Figure 7.1: The histogram of the probability to reformulate query *'flawless'* in 2013 with one month granularity.

We propose an unsupervised approach for detecting drifts in user satisfaction for pairs $\langle Q, SERP \rangle$ by applying a concept drift technique [252] leveraging reformulation signal. Concept drift primarily refers to an online supervised learning scenario when the relation between the input data and the target variable changes over time [80]. Furthermore, the reformulation signal is considered to be less noisy and if reformulations are fresh and done only by users' initiative then we can say that a reformulation signal is not biased by information coming from the search engine. Moreover, the proposed method produces:

- A list of *drift terms*, that users apply to reformulate queries when a drift happens. This list can be utilised for time sensitive query auto-completion [212].
- A list of $URLs$, that users mostly click on after reformulating initial queries. This list can be used for a recency ranking [66, 113, 114].

Our third concrete research question is:

> **RQ 7.3:** *How effective is our approach on a realistic sample of traffic of a major Internet search engine?*

We validate our approach to detect changes in user satisfaction on six months of search logs of a major commercial search engine[1]. We run our algorithm on massive transaction logs to detect pairs of $\langle Q, SERP \rangle$ where drifts happen. We analyse accuracy of detection of drift terms and clicked $URLs$.

The specific contributions of this chapter include:

[1] In this chapter we use data from `bing.com`

1. A definition of query reformulation signal as an effective way to detect an alteration in user satisfaction.

2. An analysis and formulation of a drift detection in user satisfaction.

3. An unsupervised method for detection changes in user satisfaction.

4. A large scale evaluation over real user queries, showing a high accuracy of the proposed method.

The remainder of this chapter is organised as follows. Section 7.2 describes background and related work. A formal description of the proposed method to detect changes in user satisfaction is presented in Section 7.3. Section 7.4 describes a research methodology for a large scale exploratory analysis of real user behavior logs from a commercial search engine. In Section 7.5 we describe obtained results. In Section 7.6 we discuss potential applications, that can benefit within proposed method, are described. We summarize our findings, discuss possible extensions of the current work and conclude in Section 7.7.

## 7.2 Background and Related Work

Our work examines how to model and detect changes in user satisfaction that can be a useful feature for a dynamic ranking. Huffman and Hochster [109] observed a strong correlation between the relevance of results and user satisfaction using navigational and non-navigational queries. Relevance is a complex concept (for a detailed review see [198], [197]). In a simplified view relevance *Rel* can be defined as a score for a pair of query $Q$ and document $D$, where $D$ in this case is a link $URL$ to the web page: $Rel{=}\|\langle Q, URL\rangle\|$. However, it is logical to assume that *Rel* have an altering nature because user preferences change due to external events and passage of time. Dong et al. [66] proposed a classifier to detect recency sensitive queries. This classifier gives a score, called *'buzzines'*, to a query $Q$ and $Q$ is considered as a breaking-news one if its final buzzines score exceeds some threshold. Moreover, recency ranking is proposed to overcome an issue with ranking time-sensitive queries. It proposes *Rel* that takes a *freshness* of a document into account. Dong et al. [66] proposed to incorporate recency features in a ranking model. The ranking function includes recency features: *(timestamp, linktime, WebBuzz, page topic)* and it gives a gain for ranking metrics. The paper [113] suggests a temporal click feature, called *ClickBuzz*, that captures a spiking interest in the pair $\langle Q, SERP\rangle$. This method helps to exploit user feedback for time-sensitive queries. The use of *ClickBuzz* in the ranking models leads to an improvement in $NDCG_5$. Our method can be considered as a supplement to recency ranking, it detects moments when drifts happen and we need to adjust our ranking function in order to produce up-to-date results.

User satisfaction has been researched extensively. User clicks are reasonably accurate on average to evaluate user satisfaction with pairs $\langle Q, SERP\rangle$ [11, 125], using click-through information. This user satisfaction scenario is successfully

applied to navigational queries. It is called *query-level satisfaction*. However, we have to take into account the fact that user clicks are biased:

1. to the page position in $SERP$ [56, 124];
2. to the quality of the page's snippet [250];
3. to the domain of the returned $URL$ [112].

Authors of the paper [14] claim that a search scenario for informational queries is different. Users can run follow-up queries if they are unsatisfied with the derived results. Reformulations can lead users to an answer. This scenario is called *task-level satisfaction* [66]. Past research proposed different methods for identifying successful sessions. Hassan et al. [97] used a Markov model to predict success at the end of the task. Ageev et al. [9] exploited an expertise-dependent difference in search behavior by using a Conditional Random Fields model to predict a search success. On the other hand, separate researches are interested in situations when users are frustrated. Feild et al. [75] proposed a method for understanding user frustration with the pair $\langle Q, SERP \rangle$. Authors gave users difficult information seeking assignments and evaluated their level of dissatisfaction via query log features and physical sensors. The authors demonstrated that the prediction model gets the highest quality when it is built based on query log features, described in the paper [239]. One type of user behavior that can be clearly associated with frustration is search engine switching. Authors of the paper [86] showed that one of the primary reasons users switched their search engine was due to dissatisfaction with the results on the $SERP$.

In our work, we consider a scenario when user satisfaction at time $t_i$ with $\langle Q, SERP \rangle$ turns into user frustration at $t_{i+1}$ with the same $\langle Q, SERP \rangle$. We associate user satisfaction using the reformulation signal. If the probability of reformulating query $Q$ was close to zero at $t_i$ and grows dramatically at $t_{i+1}$, then a change happened in user satisfaction. Our scenario corresponds perfectly to a definition of *real concept drift*. In dynamically changing and non-stationary environments, the data distribution can change over time because of the phenomenon of *concept drift* [202, 241]. The *real* concept drift refers to changes in the conditional distribution of the output (i.e., target variable) given the input (input features), while the distribution of the input may stay unchanged. Formally *concept drift* between time point $t_i$ and time point $t_{i+1}$ can be defined as [80]:

$$\exists X : P_{t_i}(X, y) \neq P_{t_{i+1}}(X, y), \tag{7.1}$$

where $P_{t_i}$ denotes the joint distribution at time $t_i$ between the set of input variables $X$ and the target variable $y$. In this work, we follow a lead of [32, 69, 104] and reuse methods from supervised machine learning and statistical learning theory to design and analyze suitable statistics for drift detection.

Changes in data distribution over time may manifest in different forms, as illustrated in Figure 7.2. The presented types of concept drift are perfectly aligned to the reformulation signal:

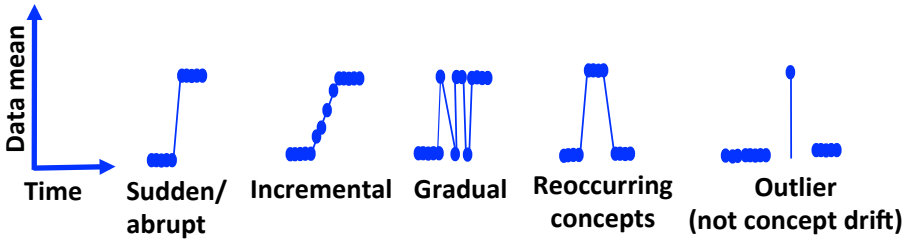



Figure 7.2: Patterns of changes over time [80].

- A drift may happen *suddenly/abruptly* by switching from one concept to another, that may correspond to breaking-news queries such as *'nelson mandela'* (issued the day of his death).
- A drift can be *incremental*, e. g. a query *'cikm conference'* may drift each year, queries referring to a new model of a device may cause an incremental drift: users incrementally move their preferences from *'windows phone 7'* to *'windows phone 8'*.
- A drift can be *gradual*, e. g. relevant new topics change from *dwelling to holiday homes*, while the user does not switch abruptly, but rather keeps going back to the previous interest for some time.
- A drift can be *reoccurring*, e. g. seasonal queries: '30 % cvs coupon'→'30 % cvs coupon **black Friday**'.
- One of the challenges for concept drift handling algorithms is not to mix the true drift with an outlier or noise which refers to a once-off random deviation or an anomaly. In our case, it may be spam queries. We will show an example of an outlier in Section 7.5.4.

To summarize, the key distinctions of our work compared to previous efforts are: a clear and well-defined approach to detecting changes in user satisfaction using the reformulation signal; an in-depth analysis of changes is searchers behavior that results in accurate detection of drifts in user satisfaction. Moreover, our framework works in an unsupervised manner, it does not require any labelling.

## 7.3 Detecting changes in user satisfaction

In this section we present an overview of a developed framework for detecting changes in user satisfaction due to some external events and overtime decay. Our framework uses the growth of the reformulation signal as an indication of user dissatisfaction with the pair $\langle Q, SERP \rangle$. In other words, if the probability to

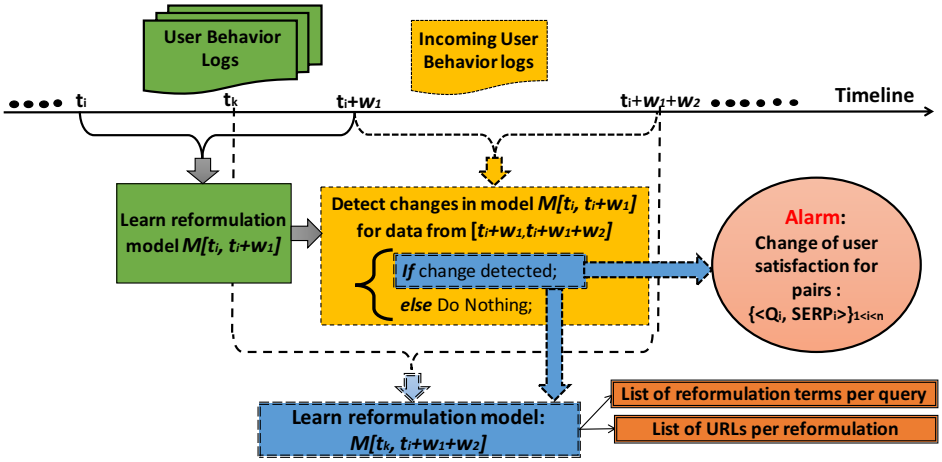



Figure 7.3: Overview of a framework for detection changes in user satisfaction with search results.

reformulate a query $Q$ to $Q'$ grows dramatically then users are no longer satisfied with the pair $\langle Q, SERP \rangle$. The desired results for the query $Q$ have been changed and now users expect to derive $SERP'$ as the answer for $Q$.

The proposed framework monitors user interactions and it triggers an alarm to the system when changes happen. Moreover, if indicated, our framework can produce the following additional output per a query:

- a list of drift terms, which users added to reformulate the query $Q$;
- a list of drift $URLs$, which users clicked on after issuing $Q'$.

A detailed diagram of our framework is presented in Figure 7.3. In following sections we will describe our framework in details:

1. how do we construct *user behavioral logs* (Section 7.3.1);
2. how to model the reformulation signal (Section 7.3.2);
3. how to detect drifts in the reformulation signal in an unsupervised manner (Section 7.3.3).

### 7.3.1 Creating User Behavioral Logs

In this section, we describe how to derive *user behavioral logs* (in Figure 7.3) from search interaction logs.

We consider the following scenario: we have a stream of queries submitted to a search engine. In response to each query, the engine returns $SERP$. Users may

decide to click on one or more $URLs$ in $SERP$, reformulate their queries, or end their sessions. These types of user interactions are stored in search interaction logs.

We convert standard search interaction logs to the user behavioral logs where we store information only about reformulations of issued queries. We use a query expansion definition from [108, 118] to detect terms which users used for reformulations.

An example of user behavioral entries is presented in Figure 7.4. Each entry consists of four columns:

- **Session ID** is a session identification information;
- **Timestamp** is a time when an action is performed;
- **Action** is an action type, that a user performed: we record the following action types: search, reformulation, and click on a $SERP$ page.
- **Action details** are details of a user's action: for the search action we record an issued query, for the click action we record an identifier of clicked page, for the reformulation action we record a reformulation term.

For example **'2014'** is a *reformation term* for the initial query *'cikm conference'* if users are looking for up-to-date information about the conference.

| ***Session Id*** | ***Timestamp*** | ***Action*** | ***Action details*** |
|---|---|---|---|
| ..... | | | |
| 123457 | 1388494920 | search | Query ='flawless' |
| 123457 | 1388494980 | click | Page Id = '755' |
| 123457 | 1388495060 | reformulation | Query ='flawless beyonce' => Reformulation = 'beyonce' |
| 123457 | 1388495115 | click | Page Id = '170' |
| 123458 | 1388495415 | search | Query ='cikm conference' |
| 123456 | 1388361661 | reformulation | Query ='cikm conference' => Reformulation = '2014' |
| 123456 | 1388361720 | click | Page Id = "45" |
| ..... | | | |

Figure 7.4: An example of user behavioral log.

User behavioral logs are suitable to collect a dictionary of the reformulation terms:

$$D_Q = \{K_j\}_{j=1}^{n}, \tag{7.2}$$

where $K_j$ is $j^{th}$ the reformulation term used to change the query $Q$, $n$ is the number of the reformulation terms used for expanding the query $Q$.

In the next section, the dictionary of the reformulation terms will be utilized for modelling the reformulation signal.

### 7.3.2 Modelling the Reformulation Signal

In this Section, we address our **RQ 7.1:** *What behavioral signal can be used to infer changes in user satisfaction?* We describe how to build a reformulation signal model, that is presented in our framework in Figure 7.3 as *'Learn Reformulation Signal'*.

We build the reformulation signal ($RS$) of queries for a time period $[t_i, t_i+w_1]$, using the user behavioral logs. $RS$ of the query $Q$ would be:

$$RS = \left\{P_{[t_i,t_i+w_1]}\left(K_j, Q\right)\right\}_{j=1}^{n}, \tag{7.3}$$

where $w_1$ is the selected size of the inference window, $P\left(K_j, Q\right)$ is a joint distribution of the query $Q$ and its reformulation term $K_j$ during the time period $[t_i, t_i + w_1]$.

When time $(t_i + w_1 + w_2)$ comes we rebuild the reformation signal of $Q$ for the time period $[t_i + w_1, t_i + w_1 + w_2]$ using Equation 7.3:

$$RS = \left\{P_{[t_i+w_1,t_i+w_1+w_2]}\left(K_j, Q\right)\right\}_{j=1}^{m}, \tag{7.4}$$

where $w_2$ is the size of a test window.

The presented model for the reformulation signal will be used to detect changes in user satisfaction in the next section.

### 7.3.3 Detecting a Drift in Reformulation Signal

In this section, we investigate our **RQ 7.2:** *How can we detect changes in user satisfaction using reformulation signal?* We present an algorithm for detecting a drift in user satisfaction using the reformulation signal. Our goal is to detect statistically significant changes. This action is depicted in Figure 7.3 as *'Detect Changes'*.

Let us introduce a definition of *a drift in the reformulation signal* between two periods at time $[t_i, t_i + w_1]$ and $[t_i + w_1, t_i + w_1 + w_2]$ using Equations 7.3 and 7.4:

$$\exists Q' : P_{[t_i,t_i+w_1]}\left(K_j, Q\right) \neq P_{[t_i+w_1,t_i+w_1+w_2]}\left(K_j, Q\right), \tag{7.5}$$

where $P_{[t_i,t_i+w_1]}\left(K_j, Q\right)$ denotes the joint distribution of query $Q$ and its reformulation term $K_j$ at the time period $[t_i, t_i + w_1]$ .

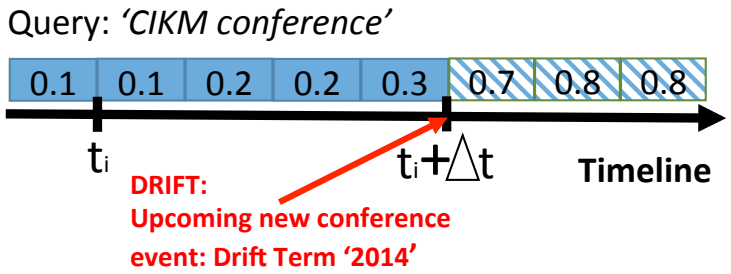



Figure 7.5: Example of concept drift in probability to reformulate the query *'CIKM conference'* using drift term *'2014'*.

It is important to determine what it means when the distribution has changed. If the drift in the reformulation signal is statistically significant, then *we assume that user satisfaction with the following pair has decreased dramatically*:

$$\langle Q_{[t_i+w_1,t_i+w_1+w_2]}, SERP_{[t_i,t_i+w_1]} \rangle, \tag{7.6}$$

where $Q_{[t_i+w_1,t_i+w_1+w_2]}$ is the query issued at the time period $[t_i + w_1, t_i + w_1 + w_2]$; $SERP_{[t_i,t_i+w_1]}$ is search results, that were generated for $Q$ at the time period $[t_i, t_i + w_1]$, and it is still shown during the time period $[t_i + w_1, t_i + w_1 + w_2]$.

However, users are no longer satisfied and they reformulate $Q$ using some drift term $K_j$. The fact that the drift has happened at time $(t_i + w_1 + w_2)$ can be a signal that we need to generate a new $SERP_{[t_i+w_1,t_i+w_1+w_2]}$ for $Q$ to improve user satisfaction.

Let us consider an example of a drift in the reformulation signal for the query *'cikm conference'* in Figure 7.5. Users were satisfied with $SERP$ that was returned by the query *'cikm conference'* at time $t_i$. However, at time $t_i + \Delta t$ a probability to reformulate the query has been changed dramatically and the term *'2014'* is the most frequent reformulation. Most likely users have changed their behavior due to an upcoming conference event and they could not find the right link in $SERP$ that was optimized for clicks from the last year.

The proposed algorithm $DDSAT$ for detecting changes in the query reformulation signal to discover changes in user satisfaction is presented in Algorithm 3. We will explain how $DDSAT$ works next.

Let us clarify which an input and an output $DDSAT$ has:

- ***DDSAT* Input.** Our current implementation of the algorithm is using a fixed size of the inference window $w_1$ that equals to one month. We also experiment with two sizes of the test window $w_2$: two weeks and one week. We calculate an error threshold $e$, using the following formula as described in [32]:

$$e = \sqrt{\frac{1}{2m} * \sigma_W^2 * \ln\frac{4}{\delta'}} + \frac{2}{3m}\ln\frac{2}{\delta'}, \tag{7.7}$$

where $m$ is the harmonic mean of $||w_1||$ and $||w_2||$, $\sigma^2_W$ is the observed variance of the elements in window $W = w_1 \cup w_2$ and $\delta' = \frac{\delta}{n}$, $\delta$ is a confidence value and $n$ is a total size of two windows. For experimentation, we run our algorithm with the three different confidence values: $\delta = \{0.05, 0.1, 0.3\}$.

- ***DDSAT* Output.** *DDSAT* returns an alarm as an output if the drift happens.

Let us consider the method $processDetectedConceptDrift()$ in Algorithm 3 that deals with the detected drifts. Moreover, the function $processDetectedConceptDrift()$ has two additional input parameters which show how the observed drifts influence the current system:

- **Parameter 'extra'** is a boolean variable, if it is *'true' DDSAT* will produce two extra statistics:
  1. a list of drift terms, which can be used for serving a fresh query suggestion;
  2. a list of drift $URLs$ which can be used for reranking of $SERP$.
- **Parameter 'update'** is a boolean variable, if it is *'true' DDSAT* will update the reformulation signal.

**Algorithm 3** Algorithm for detection drift in user satisfaction using the reformulation signal ($DDSAT$)

**Require:** inference window $w_1$;
test window $w_2$;
error threshold $e$;
start time $t_i$;
produce extra information $extra \leftarrow true, false$;
learn new reformulation signal $update \leftarrow true, false$;
$U$ser $B$ehavioral $L$og ($UBL$);
**Ensure:** $drift \leftarrow true, false$
1: $\{RS_{w_1}(Q_k)\}_{m=1}^{k} \leftarrow$buildRefSignal$(UBL_{[t_i,t_i+w_1]})$
2: $\{RS_{w_2}(Q_k)\}_{m=1}^{k} \leftarrow$buildSignal$(UBL_{[t_i+w_1,t_i+w_1+w_2]})$
3: **for** $Q_m, K_j \in ULB$ **do**
4: **if** $|\mu(P_{w_1}(K_j, Q_m)) - \mu(P_{w_2}(K_j, Q_m))| > e$ **then**
5: $drift \leftarrow true$
6: $processDetectedConceptDrift(extra, update)$
7: **else**
8: $drift \leftarrow false$
9: **end if**
10: **end for**
11: **return** $drift$

The presented algorithm *DDSAT* works in an unsupervised way, it does not require any human labelling. It can be shown that *DDSAT* has a linear complexity.

The proposed framework can be used as a monitoring tool, which alarms when user satisfaction changes for a particular pair: $\langle Q, SERP \rangle$. Our framework also gives an explanation for detected changes, it returns the list of drift terms. Moreover, it suggests a possible solution for serving up-to-date $SERP$ and returns the list of the most frequently clicked $URLs$ after reformulating $Q$ using the drift terms.

## 7.4 Experimental setup

The ultimate goal of the presented framework is to detect changes in user satisfaction. To answer our **RQ 7.3:** *How effective is our approach on a realistic sample of traffic of a major Internet search engine?* we experiment with the search interaction logs of the commercial search engine – `bing.com`. We run our framework over this dataset. In *DDSAT* algorithm we set to *'true'* the parameters: *'extra'* and *'update'* of the method *processDetectedConceptDrift()*. The extra statistics are used to setup a human assessment. Our evaluation scenario is described in Section 7.4.2. The final results are presented in Section 7.5.

### 7.4.1 Data

Our experimental data comprises of search interaction logs of the commercial search engine that were collected during six months: *September* 2013*, October* 2013*, November* 2013*, December* 2013*, January* 2014*, February* 2014. We only include log entries for US-based traffic. We derive user behavioral logs from the selected search interaction logs as presented in Section 7.3.1. Each month of data consists of over 10 million records.

### 7.4.2 Evaluation Methodology

In this section we describe how we organize evaluation of our framework results:

1. the derived list of detected changes with drift terms;
2. the derived list of the most clicked $URLs$ per drift.

First, let us present a format of the presented system output that needs to be evaluated. Our framework returns results in the form presented in Figure 7.6. It contains:

- **Date -** is a time period when drift happened;
- **Initial Query -** is a query that users issued;
- **Drift term -** is a term that cause the drift because users added it to expand the initial query at time period depicted as *Date*.

- **URL -** is a link that users clicked the most after issuing the reformulation.

| *Date* | *Initial query* | *Drift Term* | *URL* |
|---|---|---|---|
| ..... | | | |
| Oct. 2013 | novak djokovic | fiancee | $URL_{48}$ |
| Oct. 2013 | CIKM conference | 2014 | $URL_{44}$ |
| Jan. 2014 | flawless | beyonce | $URL_{578}$ |
| Jan. 2014 | feliz ano nuevo | 2014 | $URL_{48}$ |
| Jan. 2014 | ct 40ez | 2013 | $URL_{109}$ |
| Feb. 2014 | when is fastnacht day | 2014 | $URL_{48}$ |
| Feb. 2014 | mormons olympics | 2014 | $URL_{409}$ |
| ..... | | | |

Figure 7.6: Example of an output of the framework. Column *URL* is anonymized.

### Human Drift judgments

For the evaluation, a group of annotators were instructed to exhaustively examine the detected drift terms. The judges were well trained to understand time-related and event-related drifts in user behavioral data. They were given relevant examples of drift, e. g.:

1. The latest Olympic games were held in February 2014 and users run related queries such as *'medals olympics'* (users were looking for information about medals among USA 2014 Olympic team). However, if users were served with results from Olympics 2012 then they had to reformulate the query using the reformulation term *'2014'* in order to find desired results. That is an example of a gradual drift, because users got interested in 2014 Olympics gradually over two years.

2. Another example would be a breaking news query such as *'novak djokovic'* (a Serbian professional tennis player) who got engaged at the end of September 2013. Users were interested in this news and tended to reformulate the query by adding the reformulation term *'fiancee'*. That is an example of a sudden drift. This kind of drifts have quite a short lifetime because users remain interested during a limited period of time. However, it is important to serve it right in time.

An example of an evaluation task for the annotators is presented in Figure 7.7 (A). The judges were asked to decide: Can the term $T$ be a drift term for the

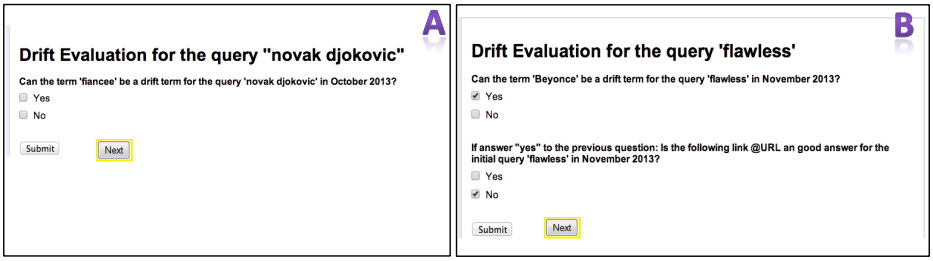



Figure 7.7: Two fragments of an evaluation task for the annotators: (A) is the task when we do not have most clicked $URL$ because clicks are diverse and (B) is the task when we can suggest $URL$ and (B).

query $Q$? The judges were allowed to use external information sources to find answers.

Every discovered drift, characterised by a drift term, is judged by three different annotators using binary classes: *'0' = wrong* and *'1'= right*. The final score is calculated based on three judgments.

We use accuracy as a final evaluation metric that we refer as *Drift Accuracy*.

### Human Judgments for Drift $URLs$

We calculate a statistic for $URLs$ which users clicked in $SERP'$ that is derived after reformulation $Q$ to $Q'$. If the probability of clicking on $URL$ is greater than 0.5, then it is drift $URL$. If clicks are diverse we cannot produce any $URL$.

Judges, whom we asked to evaluate drift $URL$, answered the following question: *'Is* $URL$ relevant *for the initial query* $Q$ *at the time period* $T$*?'* In other words, we asked human annotators to evaluate a tuple $\langle Q, URL, T \rangle$ as proposed in the paper [66].

Every discovered drift $URL$ is also judged by three different persons using binary classes. The final score is calculated based on three judgments. We use accuracy as a final evaluation metric for drift $URL$, that we refer as *URL Accuracy*.

We described the large-scale evaluation of our method based on a real dataset from the commercial search engine `bing.com` that was collected during six months. As we will show next we can precisely identify drift in user satisfaction using the reformulation signal.

## 7.5 Experimental Results

We now summarize the results for detection changes in user satisfaction using the reformulation signal to answer our **RQ 7.3:** *How effective is our approach on a*

*realistic sample of traffic of a major Internet search engine?* The proposed solution is working in an unsupervised way and can be applied to large search interaction logs. As a ground truth, we use the human judgments that are described in Section 7.4.2.

### 7.5.1 Defining Sizes of Inference and Test Windows

It is important to note that we fix a size of the inference window $w_1$ to one month of data. For the test window $w_2$, we experiment with three different sizes: one week, two weeks, one month. As a final result, we will report for $w_2$ equals two weeks.

For our experimentation we use data described in Section 7.4.1, that can be easily transformed to user behavioral logs. The algorithm *DDSAT*, proposed in Algorithm 3, is running on derived data in the following way:

1. *DDSAT* starts at time $t_i$ (for our datasets: $t_i$ equals to $1^{st}$ *of September*);
2. *DDSAT* builds the reformulation signal ($RS$) based on the time period $[t_i, t_i + w_1]$ (for our datasets: the reformulation signal is built on September 2013);
3. *DDSAT* detects drifts in $RS$ based on the time period $[t_i + w_1, t_i + w_1 + w_2]$ and produces a list of detected drifts (for our datasets: first two weeks of October 2013);
4. *DDSAT* reassigns $t_i$ to $(t_i + w_1)$ and goes to *1.*

We combine all detected drifts and drift $URLs$ and evaluate them using methodology described in Section 7.4.2.

However, for other domains the sizes of $w_1$ and $w_2$ are dependent on many aspects such as volume of a traffic, type of a served content and so on. Implementers with a domain's knowledge should decide on how often running our framework. However, we plan to extend the algorithm *DDSAT* so that it can determine on the fly when there was a drift.

### 7.5.2 Defining confidence value

We evaluate the discovered list of the drifts in user satisfaction to check which confidence level suits our needs best. The randomly selected part (30%) of human judgments for detected drifts (Section 7.4.2) are used to calculate the accuracy below. Our findings are the following:

- for the confidence value $\delta = 0.05$ accuracy is 65%
- for the confidence value $\delta = 0.1$ accuracy is 68%
- for the confidence value $\delta = 0.3$ accuracy is 66%

Table 7.1: The accuracy of the drift detection depends on the number of users who issued reformulations. The metrics are calculated based on the results obtained with the confidence value $\delta = 0.1$.

| Number of Users | Drift Accuracy |
|---|---|
| $[1000, 1300)$ | 98% |
| $[800, 1000)$ | 67% |
| $[500, 800)$ | 80% |
| $[250, 500)$ | 82% |
| $[1, 100)$ | 66% |

The rest of judgments (70%) are used to calculate a final accuracy for our drift detection method in Section 7.5.3.

Hence, we use the confidence value $\delta = 0.1$ for the future evaluation because it gives us the highest accuracy.

### 7.5.3 Evaluating DDSAT

In this section we describe the experiments we conducted to evaluate the accuracy of our method *DDSAT*. We evaluate two types of accuracy:

1. we present the accuracy of overall drift detection that is calculated based on 70% of human judgments collected in Section 7.4.2;

2. we demonstrate the accuracy of detected drift $URLs$ that is calculated based on annotator's judgments collected in Section 7.4.2.

#### Drift Accuracy

Drift accuracy is a percentage of times when the drift in user satisfaction is correctly detected using the reformulation signal. We calculate drift accuracy with respect to the number of users who issue the reformulation. The obtained accuracy is presented in Table 7.1.

Of course, the best result is characterized by the greatest amount of users. Rows in Table 7.1 for the number of users in a range $[800, 1000)$ and $[500, 800)$ have a lower accuracy than the accuracy rate for the numbers of users in the range $[250, 500)$ because they include smaller number of detected drifts in user satisfaction.

#### Drift $URL$ Accuracy

Drift $URL$ accuracy is a percentage of relevant $URLs$ among the list of proposed drift $URLs$. The obtained accuracy is presented in Table 7.2. The quality of derived drift $URLs$ is very high especially for the number of users greater than 250. We see in Table 7.2 the same situation as in Table 7.1 for the number of users in a range $[800, 1000)$ and $[500, 800)$. They have lower accuracy than the

Table 7.2: The accuracy of drift $URL$ depending on the number of users who issued reformulations. The metrics are calculated based on results obtained with confidence value $\delta = 0.1$.

| Number of Users | Drift $URL$ Accuracy |
|---|---|
| $[1000, 1300)$ | 100% |
| $[800, 1000$ | 81% |
| $[500, 800)$ | 85% |
| $[250, 500)$ | 91% |
| $[1, 100)$ | 87% |

number of users in the range $[250, 500)$ because they include a smaller number of detected drift $URLs$.

Our framework includes $URL$ into the list of drift $URLs$, if the probability of clicking on them after reformulating is higher than 0.5. Potentially, detected drift $URLs$ can be applied directly into a learned ranking function as a *'freshness feature'* or used for a re-ranking of current results.

To summarize our evaluation of $DDSAT$, we recommended to determine a confidence value for the drift detection that gives the highest accuracy of the detected drift. In our case, we obtained the confidence value $\delta = 0.1$. The proposed algorithm was evaluated from two points of view:

1. the accuracy of the detected drift in user satisfaction is high and it gets especially precise if the number of users who issued reformulation is greater that 250 (Table 7.1). We do not report the row '$> 1000$' because it not always realistic for smaller search engines;

2. the accuracy of how relevant detected drift $URLs$ are. It is especially accurate if the number of users who issued reformulation is greater than 250 (Table 7.2).

### 7.5.4 Detecting Anomalies in Results

In this section we show an example of *outliers* in a concept drift and how to deal with this kind of anomalies.

User behavior on Web is not always reliable, it can be sometimes spurious. It is important for the algorithm $DDSAT$ to know how to remove this anomalies from user behavioral logs data in order to return more accurate results of drift detection.

For instance, *'spurious behavior'* can be caused by Search Engine Optimization (SEO) that is a process of affecting the visibility of websites or web pages in search results of search engines. In general, SEO aims to push a site to a higher rank on the search results page, and more frequently a site appears in the search results list, the more visitors it will receive from the search engine's users. In order to achieve the goal, SEO considers how search engines work, what people search

for, the actual search terms or keywords typed into search engines and which search engines are preferred by their targeted audience. Optimizing of a website may involve editing its content, HTML and associated coding to both increase its relevance to specific keywords and to remove barriers for indexing activities of search engines.

While we were analysing the list of derived drifts we noticed abnormal drifts, e. g. the query *'aol mailbox sign in'* was reformulated using the drift term *'agnes corky'*. The reformulation was issued by more than 200 users. However, this drift did not make any sense. This behavior most probably was simulated. However, we noticed that only one click happened and that clicked page referred to the website with the domain named *'seotest'*. Hence, we concluded that it is the anomaly.

For the final results, this kind of anomalies need to be filtered out. They are removed by using the following heuristic rule: *'If the number of users who issued reformulation:* $(U)$ *is much greater than the number of user clicks on* $SERP'$ *(search results after reformulating an initial query):* $(Click_U)$ *then the detected change is the Anomaly':*

$$\text{if } U \gg Click_U \text{ then } Anomaly. \tag{7.8}$$

To summarise our experimental results, the proposed technique for detecting changes in user satisfaction using the reformation signal works well on real datasets. The observed results over large datasets (all traffic from the commercial search engine `bing.com` during 6 months) are both substantially and statistically significant. Furthermore, we have shown that results of our framework, such as lists of $URLs$, can be potentially useful for ranking.

## 7.6 Applications

In this section we discuss potential applications where the results of the developed framework can be used.

### 7.6.1 Learning to Rank

A key component of our system is the algorithm $DDSAT$ that is monitoring user engagement. It alarms when changes in user satisfaction happen with the pair $\langle Q, SERP \rangle$. $DDSAT$ alarm is a signal that user intent for $Q$ drifted and we need to change $SERP$ to satisfy changes in user needs. Potentially, the detected drift can be applied directly into a learned ranking function as a *'freshness feature'* or used for re-ranking. Moreover, our framework produces a list of $URLs$, which users prefer after reformulating the initial query. This list also can be incorporated into a ranking model.

### 7.6.2 Query Auto-Completion

Query auto-completion is an important feature of online search engines that enhances search experience by saving users time which otherwise would be spent

on typing. A time-sensitive approach has been proposed in [212] for query auto-completion. Our framework also returns drift terms, which are reformulation terms that cause a drift in the reformulation signal. Hence, this list can be used for time-sensitive query auto-completion.

### 7.6.3 Automatically Detecting Under-performing Queries

Automatic detection of problematic queries, where search engines do not return a required result and users are dissatisfied with their search results, has been extensively studied [9, 75, 97, 134]. However, previous work largely utilises user interaction features, topical and lexical attributes to detect such underperforming queries. Time-sensitive nature of user satisfaction has not been considered.

In this chapter, we propose the method to identify drifts in user satisfaction over time. The proposed framework monitors a system and it signals an alarm when drift in user satisfaction with the pair $\langle Q, SERP \rangle$ happens. Hence, when we know a problematic query we can retrain our ranker in order to improve quality of retrieved $SERP$. We can use the engagement on the reformulated query in order to derive training pairs.

## 7.7 Conclusions

In this chapter, our main research question in this chapter was: *How can we detect a drift in user satisfaction with the pair $\langle Q, SERP \rangle$ using users' interactions on the $SERP$?*

First, we studied **RQ 7.1:** *What behavioral signal can be used to infer changes in user satisfaction?* We explored the utility of incorporating the query reformulation signal in detecting changes in user satisfaction.

Next, we investigated **RQ 7.2:** *How can we detect changes in user satisfaction using reformulation signal?* We leveraged the concept drift techniques to detect changes in user satisfaction with the pair $\langle Q, SERP \rangle$ over time due to some events. The appearance of a drift requires a modification of the $SERP$ to satisfy shifted user needs. We introduced a novel *Drift Detection in user SATisfaction (DDSAT)* algorithm, that accurately detects changes. The proposed algorithm works in an unsupervised manner, it does not need any labelled data. *DDSAT* is a part of the developed framework for detecting changes in user satisfaction. The algorithm of the drift detection in user satisfaction which we presented in this chapter can be incorporated in many search-related applications where freshness is required, e. g. in recency ranking and query auto-completion.

Finally, we analyzed the prediction quality of DDSAT to answer **RQ 7.3:** *How effective is our approach on a realistic sample of traffic of a major Internet search engine?* We conducted a large-scale evaluation using data from the commercial search engine `bing.com`. The dataset was collected during six months. Our experiments show that the algorithm *DDSAT* works with a high accuracy. Moreover, our framework outputs the list of drift terms and the list of $URLs$, which can be used for the future re-ranking of $SERP$.

Our general conclusion is that the drifts in reformulation signal reflect changes in user satisfaction with $\langle Q, SERP \rangle$.

We believe that the current implementation of the algorithm *DDSAT* can be improved. The algorithm uses the fixed sizes of the inference and test windows. However, it is not always suitable. For instance, the size of the test window for the sudden drift can be way shorter compared to incremental drifts. We anticipate that the size of the test window should be proportional to the reformulation frequency. We would like to develop a method to identify dynamically the size of the inference and test windows as future work. We also would like to identify the type of the detected drift in Figure 7.2. It is important to know in order to define the lifetime. If our algorithm detects the sudden drift (e. g. breaking news queries) then its lifetime is much shorter compared to incremental or sudden drifts. We would like to develop a method to identify automatically the type of the drift.

# 8 Failed SERPs

Web search is always in a state of flux: queries, their intent, and the most relevant content are changing over time, in predictable and unpredictable ways. Modern search technology has made great strides in keeping up to pace with these changes, but there remain cases of failure where the organic search results on the search engine result page (SERP) are outdated, and no relevant result is displayed. Failing SERPs due to temporal drift are one of the greatest frustrations of web searchers, leading to search abandonment or even search engine switch. Detecting failed SERPs timely and providing access to the desired out-of-SERP results has huge potential to improve user satisfaction. Our main findings are threefold: First, we refine the conceptual model of behavioral dynamics on the web by including the SERP and defining (un)successful SERPs in terms of observable behavior. Second, we analyse typical patterns of temporal change and propose models to predict query drift beyond the current SERP, and ways to adapt the SERP to include the desired results. Third, we conduct extensive experiments on real world search engine traffic demonstrating the viability of our approach. Our analysis of behavioral dynamics at the SERP level gives new insight in one of the primary causes of search failure due to temporal query intent drifts. Our overall conclusion is that the most detrimental cases in terms of (lack of) user satisfaction lead to the largest changes in information seeking behavior, and hence to observable changes in behavior we can exploit to detect failure, and moreover not only detect them but also resolve them.

## 8.1 Introduction

The information seeking behavior of users on the web is inherently sensitive to changes happening in world [157, 185]. As the web reflects the world around us, content is changing in predictable and unpredictable ways, affecting the search intent and queries issued by users. Added to that, searchers express their complex information needs in short queries, causing an inherent ambiguity in their statements of request: the query intent is specific to the context of the user and

the point in time. It is a formidable achievement of modern search engines that they manage to keep up to pace with changing content, at equally formidable costs in crawling and updating search engines indexes. In particular, for updating rankers, click through information in interaction logs are crucial [11, 124]

Yet, there remain cases of failure where the organic search results on the search engine result page ($SERP$) are outdated, and no relevant result is displayed. This can be caused by temporal query intent drift, where the desired pages for a query are changing over time, and the historical transaction logs privilege the outdated results. For example, if users were searching for *'Malaysia airlines flight'* in March 2014 they most likely wanted to see news about the *Malaysian flight 370* that disappeared. However, if users issued the same query in July 2014 they mostly likely were searching for information about the *Malaysian flight 17* that is presumed to be shot down. Figure 8.1 shows daily Wikipedia page views for the MH17 and MH370 pages over 2014, with striking increases from 0 to 100s of thousands of page views when the events happened.

The Malaysian Airlines example can be characterized as a "sudden" drift which may cause the $SERP$ to become outdated. Such changes can be associated with the news, and received the most attention in research community [66, 67]. However, changes may happen over a longer period of time and not necessarily bring an increase in the volume of traffic. For instance, if users issued the query *'CIKM conference'* in 2014 they were satisfied with results referring to the page `http://cikm2014.fudan.edu.cn/` and this page got a majority of clicks. However, the conference site has been changed and the same query issued in 2015 should be linked to the different page `http://www.cikm-2015.org/`. The CIKM example can be characterized as an "incremental" drift where the intent of the original query is changing over a longer period of time.

In this chapter, we examine a generic approach to detect $SERPs$ that become out of sync with the query intent. Specifically, users issue a query $Q$ and a search engine returns search result page ($SERP$) that is a ranked list of URLs:

$$SERP = (url_1, \ldots, url_i, \ldots, url_n).$$

Our users are expected to click on some $url_i$ on the $SERP$ that satisfies their information need, and the order of URLs on the $SERP$ is based on various features and optimized to fit a history of user interactions with a pair $\langle Q, SERP \rangle$. As a result, the $\langle Q, SERP \rangle$ shown at a given point in time will reflect the user preferences over an earlier period of time. However, this gives no guarantee on the quality of the current $\langle Q, SERP \rangle$ as user preferences are sensitive to time and events happening in the world. We aim to detect cases of $SERP$ failure due to a significant drift in query intent over time.

Our aim is to detect failed $SERPs$ due to intent drift in an unsupervised way not relying on signals from other sources than the web traffic, language independent and not relying on rules or templates, independent of volume capturing both head and tail query drift. Hence we use behavioral signals as indicators of user (dis)satisfaction, such as click-through information [11, 125] and in particular query reformulations [9, 96, 145]. Specifically, in this chapter, we are trying to answer the following main research question:

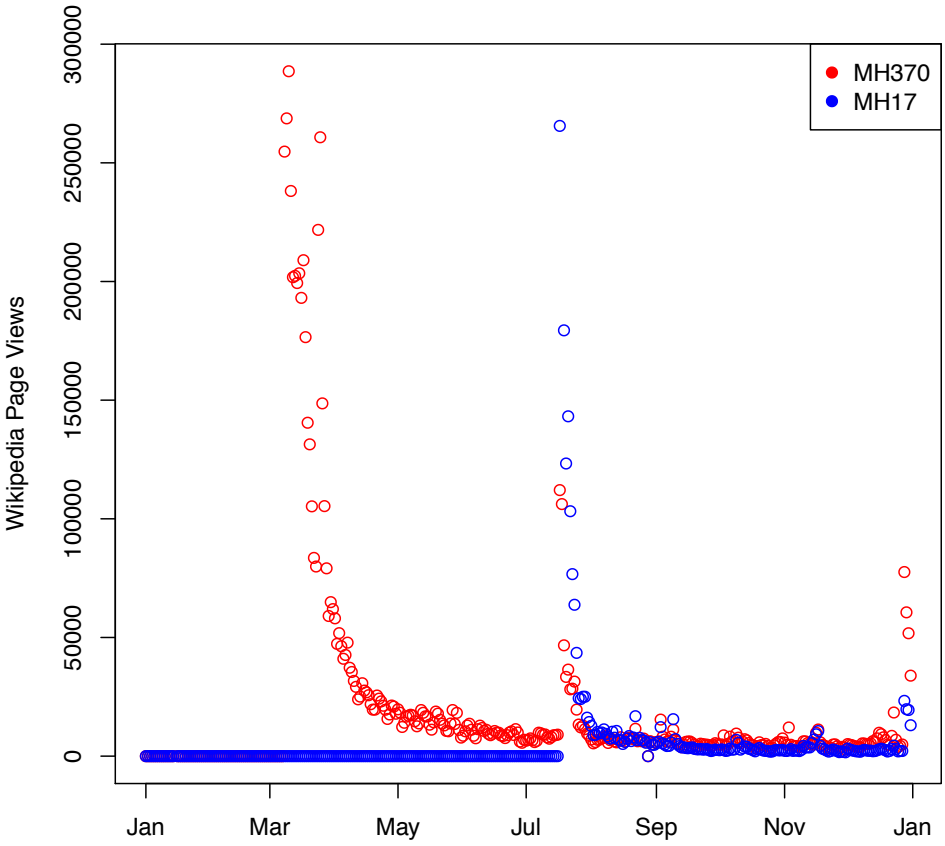



Figure 8.1: Wikipedia page views per day over 2014 for `https://en.wikipedia.org/wiki/Malaysia_Airlines_Flight_17` and `https://en.wikipedia.org/wiki/Malaysia_Airlines_Flight_370`.

> *By analyzing behavioral dynamics at the SERP level, can we detect an important class of detrimental cases (such as search failure) based on changes in observable behavior caused by low user satisfaction?*

We break up the main research problem into three different parts. Our first concrete research question is:

> **RQ 8.1** *How to include the SERP into the conceptual model of behavioral dynamics on the web? How to identify (un)successful SERPs in terms of drastic changes in observable user behavior?*

We conduct a conceptual analysis of behavioral dynamics from the *SERP*'s perspective, and introduce failure and success at the SERP level, analyzing their behavioral consequences identifying indicators of success and failure. We then analyze success and failure in the light of changing query intents over time, and

identify an important case of $SERP$ failure due to query intent drift, and suggest an approach to detect a failed $SERP$ due to query intent drift by significant changes in behavioral indicators of failure.

Our second concrete research question is:

> **RQ 8.2** *Can we distinguish different types of SERP failure due to query intent drift (e. g., sudden, incremental), and when and how should we update the SERP to reflect these changes?*

We study different types of possible query intent drift inspired by the literature on concept drift [79]: sudden, incremental, gradual and reoccurring. It is important to be able to classify the type of changes in user satisfaction, because a sequence of actions a search engine should perform to normalize a situation can be different. We identify relevant parameters, such as the window of change, volume or popularity of queries, and relevant behavioral indicators, such as the probability of reformulation, abandonment rates, and click through rates. For the two main categories of intent drift, we define an unsupervised approach to detect failed $SERPs$. We also show how the detected changes can be used to improve a ranking of search results.

Our third concrete research question is:

> **RQ 8.3** *How effective is our approach on a realistic sample of traffic of a major Internet search engine?*

We validate our approach on twelve months of search interaction logs of a major commercial search engine. We run a simplified version of our algorithm on a massive transaction log, and detected pairs of $\langle Q, SERP \rangle$ suspected of failing due to drifting query intents. We investigate the accuracy of drift detection and the accuracy of the clicked URLs of the revision to include on the $SERP$ of the original query. We look at the effectiveness of our approach for both sudden and incremental changes in query intent, by varying the duration of the window to detect failed $SERPs$.

The remainder of this chapter is organized as follows. Section 8.2 introduces earlier work on behavioral dynamics on the web, and behavioral indicators of user satisfaction focusing on the $SERP$ level. Then, Section 8.3 introduces the concept of $SERP$ success and failure, and outlines behavioral cues for their detection $SERP$ becoming out of sync over time. Followed by Section 8.4 zooming in on different types of query drift causing failed $SERPs$, and outlining practical ways of detecting them. Finally, Section 8.5 reports on extensive experiments demonstrating the real-world utility of our approach.

## 8.2 Background and Related Work

In this section we will study related work, focusing on research on topic and concept drift, on the behavioral dynamics of the web, and on user satisfaction signals on the $SERP$ level.

### 8.2.1 Topic and Concept Drift

Topic or query drift has been studied for long in IR, usually in the context of evolving information needs as may happen in routing tasks [15], or the opposite negative effect of retrieving off-topic documents lower in the ranking [213]. In particular in adaptive filtering, topic models are continuously updated when new data comes available [19]. The focus is on a general topic or standing profile that monitors a stream of data and selects relevant documents. Our focus is on the SERP, serving results to a population of users with subtle or less subtle variation in query intent, taking changes in the query intent into account over time.

Topic drift is distinct from concept drift [79, 202, 241] which, in a machine learning setting, refers to changes in the conditional distribution of the output (i. e., target variable) given the input (i. e., input features), while the distribution of the input may stay unchanged. We will use a concept drift approach in the next sections, to model changes in features indicating lack of search satisfaction, and for determining thresholds for drift detection.

### 8.2.2 Behavioral Dynamics

The changes in query popularity over time have been studied extensively in prior work. Moreover, researchers have also examined the relationship between query behavior and events [184]. There are algorithms for identifying queries that are related to breaking news and for blending relevant news results into core search results [66, 164].

Prior work on behavioral dynamics is based on three factors: (1) on changes in query dynamics and in this case authors are concentrated on the 'head' queries [157, 185, 186, 209]; (2) on changes in web content dynamics and user interaction with dynamic content [157]; and (3) how information about changes can be used:

- to improve the ranking on the $SERP$ [57, 66, 67, 164]; and
- to improve query auto-completion [212].

Additionally, Kulkarni et al. [157] explored how queries, their associated documents, and the intents corresponding to the queries change over time. Radinsky et al. [186] have done an extensive study how time-series analysis methods can be applied to predict dynamics on the web. Shokouhi [209] proposed using time-series decomposition techniques for identifying seasonal queries.

In summary, the prior studies cited above examine how general changes in content, specific content features, or query volume can be used to improve web search experience. Although much has been done to understand user web search behavior over time, few efforts have sought to construct underlying models to understand changes $\langle Q, SERP \rangle$ and even used to automatically fix the observed problems. We present the construction of models for behaviors over time, that can explain observed changes in user satisfaction with $\langle Q, SERP \rangle$.

### 8.2.3 User Satisfaction

User satisfaction with the $SERP$ has been researched extensively. It is widely adopted as a subjective measure of search experience. User clicks are reasonably accurate on average to evaluate user satisfaction with pairs $\langle Q, SERP \rangle$ [11, 125], using click-through information. This user satisfaction scenario is successfully applied to navigational queries. It is called *query-level satisfaction*. However, we have to take into account the fact that user clicks are biased:

1. to the page position in the $SERP$ [56, 124];

2. to the quality of the page's snippet [250]; and

3. to the domain of the returned URLs [112].

Al-Maskari et al. [14] claim that the search scenario for informational queries is different. Users can run follow-up queries if they are unsatisfied with the derived results, and reformulations can lead users to the desired information. This scenario is called *task-level satisfaction* [66]. On the one hand, earlier research proposed different methods for identifying successful sessions. Hassan et al. [97] used a Markov model to predict success at the end of a task. Ageev et al. [9] exploited an expertise-dependent difference in search behavior by using a Conditional Random Fields model to predict a search success. On the other hand, separate researches are interested in situations when users are frustrated. Feild et al. [75] proposed a method for understanding user frustration with the pair $\langle Q, SERP \rangle$ based on query log and physical sensor features. Kiseleva et al. [145] showed how to automatically detect changes in user satisfaction using the reformulation signal.

Earlier, White and Dumais [239] gave users difficult information seeking assignments and evaluated their level of dissatisfaction via query log features and physical sensors. They demonstrated that the prediction model gets the highest quality when it is built based on query log features. One type of user behavior that can be clearly associated with frustration is search engine switching. Guo et al. [86] showed that one of the primary reasons users switched their search engine was due to dissatisfaction with the results on the $SERP$. A recent study [122] shows a method to predict finer-grained, graded satisfaction levels. This chapter significantly extends earlier work [145], that was presented in Chapter 7, analyzing behavioral dynamics at the $SERP$ level, and explaining how and why the changes are happening. In this work we propose a methodology to define a type of changes in user satisfaction and how this information can be used to improve a ranker.

Summarizing, in this section, we presented an overview of prior work on behavioral dynamics and user satisfaction on the web, with a special focus on the $SERP$ level. In the rest of the chapter, we will study variations in user satisfaction with $\langle Q, SERP \rangle$ pair over time, starting with a conceptual analysis of success and failure at the $SERP$ level in the next section.

## 8.3 Success and Failure at the SERP

In this section we will study **RQ 8.1** *How to include the SERP into the conceptual model of behavioral dynamics on the web? How to identify (un)successful SERPs in terms of drastic changes in observable user behavior?*

### 8.3.1 (Un)successful SERPs

We first introduce the notions of successful and unsuccessful SERPs as a conceptual model. Recall from the above that we looks at the pair $\langle Q, SERP \rangle$, with a *query* $Q$ and a *search engine result page* ($SERP$) consisting of a ranked list of URLs in response to query $Q$. That is,

$$SERP_q = (url_1, \ldots, url_i, \ldots, url_n).$$

Let us further assume that queries are issued for a purpose and that the *intent* of query $Q$ can be represented as a non-empty set of desired pages $INTENT_q$. For example, conceptually speaking, a navigational query will have a singleton set $INTENT_q$, and an informational query will have a larger set of desired pages. Over a population of users there may be a distribution of intents, each giving rise to a different set of desired pages, and it is straightforward to incorporate this into the conceptual model, but for simplicity and clarity we use a single set of desired pages here.

We define a successful and unsuccessful $SERPs$ in the following way:

**Definition 6.** *(a) A $SERP_q$ is a* successful SERP *for query $Q$ if and only if $\exists url_q \in INTENT_q$ such that $url_q \in SERP_q$.*

*(b) A $SERP_q$ is a* failed SERP *for query $Q$ if and only if $\forall url_q \in INTENT_q$ such that $url_q \notin SERP_q$.*

A user issuing query $Q$ may respond to the $SERP$ in different ways. One of the possible scenarios of user interaction with the $SERP$, which is widely studied, is an event when users do not click on presented results. This case is called *search abandonment* that is known as a metrics of how successful a $SERP$ is. Research on search abandonment [53, 54, 65, 217] studied two primary abandonment cases: *bad abandonment* indicating user frustration and dissatisfaction; *good abandonment* suggesting satisfaction without needing to click. Assume we have a successful SERP in the sense of the conceptual definition above, and observe no clicked result, this suggests a case of good abandonment. Good abandonment is quite common in modern search engines because direct answers such as weather and stock quotes are returned for queries with explicit intent. Moreover, snippets can also satisfy users' information needs directly. However, if we assume a failed $SERP$, then a lack of clicked results suggests bad abandonment. Diriye et al. [65] report roughly equal fractions of good and bad abandonment, hence the abandonment rate is a secondary indicator of $SERP$ success or failure.

The other possible scenario is for users to interact with a retrieved $SERP$. Web search users often click on the $SERP$ and/or follow up with other queries.

Table 8.1: SERP Success and failure

| Behavior | Failed $SERP$ | Successful $SERP$ |
|---|---|---|
| No clicks | Bad abandonment | Good abandonment |
| Clicked result | $DSAT$ clicks | $SAT$ clicks |
| Revised query | Negative reformulation | Positive reformulation |

Many researchers have shown that clicks and reformulations can be used for a variety of tasks. However, clicks are usually considered to be as positive sign [11, 125] to detect user satisfaction with the pair $\langle Q, SERP \rangle$. In the conceptual model, we can distinguish between satisfaction (SAT) and dissatisfaction of clicks based on the desired pages:

**Definition 7.** *(a) A click on* $url_i \in SERP_q$ *for query* $Q$ *is a* SAT click *if and only if* $url_i \in INTENT_q$.

*(b) A click on* $url_i \in SERP_q$ *for query* $Q$ *is a* DSAT click *if and only if* $url_i \notin INTENT_q$.

It is an immediate corollary that there cannot be SAT clicks on a failed $SERP$, and that we can expect SAT clicks, but cannot exclude $DSAT$ clicks, on a successful $SERP$. A practical approximation to detect the difference in satisfaction is the use of dwell time, either with simple thresholds such as 30 seconds, or by advanced classification models [136].

Apart from consulting the results on the $SERP$, users may also decide to revise the query. Query reformulations have been used as indicator of search satisfaction [9, 96, 145]. Query revision may happen both in case of successful and unsuccessful $SERPs$. In case of the a successful $SERP$, for example after interacting with some relevant results, a user may refine her query to explore a further sub-topic or aspect of the query. In want of a better term, we call this type of revision a *positive reformulation*. In case of an unsuccessful $SERP$, our frustrated user may opt to formulate her query for example by spelling out her information need more explicitly, in the hope to arrive at a successful $SERP$. We call this type of revision a *negative reformulation*.

Table 8.1 summarizes the relation between the concept of successful and failed $SERP$ and indicators of user satisfaction such as search abandonment, (dis)satisfied clicks, or query revisions. While the detection of failed and successful $SERPs$ in practice is non-trivial, the conceptual analysis allows us to simply assume the existence of abstract concepts like the set of desired pages, and clear up the exact meaning of core concepts and their dependencies and consequences.

### 8.3.2 Behavioral Dynamics of SERP Failure

We now look in detail at the impact of changing query intent over time on the $SERP$, and how this affects the pair $\langle Q, SERP \rangle$. Specifically, we look at the

transition between a time point $t_i$ and a later time $t_{i+1}$:

$$\langle Q, SERP_q \rangle_{t_i} \rightarrow \langle Q, SERP_q \rangle_{t_{i+1}}.$$

Assume that at $t_i$ we have a successful $SERP$, hence it contains at least one page satisfying the intent of query $Q$ at that point in time. Due to a satisfaction click on a result, the ranker will reinforce the $SERP$'s content and will likely present the same organic results at $t_{i+1}$. Many queries such as navigational requests, are very stable and resulting in a successful $SERP$ at time $t_{i+1}$. However, there is also an important fraction of queries that has a changing intent due to something happening in the world, which may cause the $SERP_q$ to become unsuccessful at time $t_{i+1}$.

This requires detecting when a $SERP$ becomes out of sync due to changes in the query intent. There are of course subtle changes in query intent over time, leading to small changes in the click distribution with the $SERP$ for a $Q$ as studied in previous work for updating rankers. But these do not lead to an unsuccessful $SERP$ as defined in the chapter. Hence we aim to distinguish cases where the desired page is not part of the $SERP$, at least not part of top n of the organic ranking (e. g., the top 10 results).

There are cases when users are looking for a desired page that does not exist, either no longer exists or was not created or updated yet. For example, a newly created page with a winner or an outcome of an election, users are looking for the next version of iPhone, etc. Although even in these cases, there is usually a surrogate desired page that explains that the page doesn't exist, and may inform or speculate on the time when the information will become available.

### 8.3.3 Detecting Failed SERPs

Our analysis leads to the following scenario when user satisfaction at time $t_i$ with $\langle Q, SERP \rangle$ turns into user frustration at $t_{i+1}$ with the same $\langle Q, SERP \rangle$. In other words we aim to detect situations when at time $t_i$ users were satisfied with a pair $\langle Q, SERP \rangle$ and at some moment in time $t_{i+1}$ users are no longer satisfied with the same pair $\langle Q, SERP \rangle$, due to changes in the query intent for example due to some event happening in the world.

We consider the following types of behavior $BF_j$ on the $SERP$ as a sign of user frustration (lack of search satisfaction) with the $SERP$:

- $BF_1$: search abandonment;
- $BF_2$: query reformulation;
- $BF_3$: $DSAT$ clicks on the top-10 search results; and
- $BF_4$: $SAT$ clicks on the low ranked search results ($> 10$).

The intuition of our approach is that, over a population of users issuing a query, if we see a *sufficient* amount of negative reformulations, $DSAT$ and low-ranked clicks, and bad search abandonment, then we flag the $SERP$ as *failed*,

and use information about the ultimately clicked page to update the $SERP$ for the original query—hence avoid failure for future requests.

In order to satisfy a requirement about *sufficient* number we use the phenomenon of *concept drift* [79, 202, 241]. The *real* concept drift refers to changes in the conditional distribution of the output (i.e., target variable) given the input (input features), while the distribution of the input may stay unchanged. For our problem we can formally define *concept drift* between time point $t_i$ and time point $t_{i+1}$ as:

$$\exists BF_j : P_{t_i}(BF_j, \langle Q, SERP \rangle) \neq P_{t_{i+1}}(BF_j, \langle Q, SERP \rangle),$$

where $P_{t_i}$ denotes the joint distribution at time $t_i$ between the set of input variables $\langle Q, SERP \rangle$ and the target variable $BF_j$. The approach is explained in detail in the next section.

We not only intend to detect failure, but also to find and to inject the missing page to the $SERP$. In case of revisions with following $SAT$ clicks, or low-ranked clicks, we have a clear indication of the "missing" page and boost it's ranking so that it will surface on for the original query's $SERP$ for future users issuing the same query.

Summarizing, in this section, we introduced the concept of a successful and failed $SERP$ and analyzed their behavioral consequences identifying indicators of success and failure. We then analyzed success and failure in light of changing query intents over time, and identified an important case of $SERP$ failure due to query intent drift. This suggests an approach to detect a failed $SERP$ due to query intent drift by significant changes in behavioral indicators of failure. Our general conclusion is that more detrimental cases in terms of user satisfaction lead to larger changes in observable user behavior and hence more handles to detect them.

## 8.4 Types of Drift in User Satisfaction

In this section we will study **RQ 8.2** *Can we distinguish different types of SERP failure due to query intent drift (e.g., sudden, incremental), and when and how should we update the SERP to reflect these changes?*

### 8.4.1 Classifying Drift Type

This section proposes a method to detect the *types* of changes in user satisfaction. The detection of the type of change is important because it defines a strategy to fix a failed $SERP$ that is a result of changes in user satisfaction. We focus on two main types of the drifts: sudden and incremental, and will argue that the other types (i.e. gradual and reoccurring) can be represented as a combination of sudden and incremental types.

We use an increase in the query reformulations as a sign of user frustration with a shown $SERP$. The main criteria to distinguish between sudden and incremental drifts is the size of the testing window: (1) the sudden drift should be detected

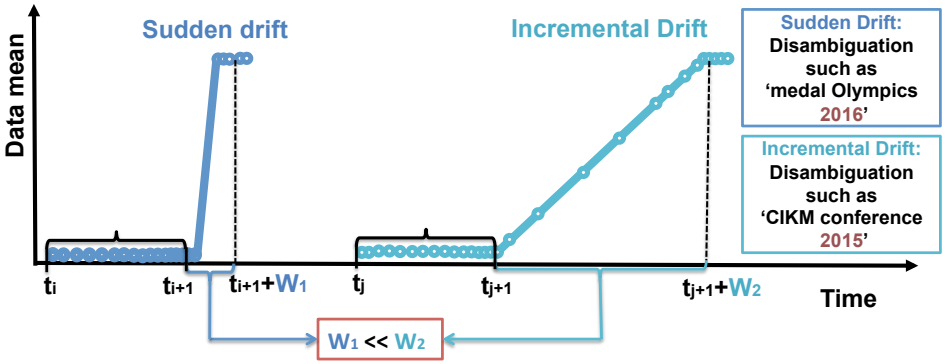



Figure 8.2: A representation of sudden and incremental types of the drifts.

during the short period of time and (2) the incremental drift can be characterized by a much longer testing period. An example is shown in Figure 8.2. Moreover, we are proposing a list of *secondary metrics* that can be used to characterize the drifts:

1. if the drift is related to the query popularity (i.e., 'head' or 'tail' queries);
2. if the volume of initial queries is changing a lot;
3. if search abandonment is observed frequently on the initial $SERP$s. In the context of an increasing number of query revisions, we will observe predominantly bad abandonment cases.

Let us characterize in details the types of changes we are studying in this work:

### Sudden change

This kind of change gets the most attention in the literature [57, 66, 157, 164, 185] because they bring a most harmful and visible effect to user experience. This drift can be characterized by a growth of query popularity over a short period of time (e. g., 'breaking news queries'), as shown on the left hand side of Figure 8.2. In order to detect sudden drifts we use a short duration of the testing window (e. g., a couple of hours until a couple of days). Using the secondary metrics the sudden drift can be defined as:

1. the sudden drift is likely concerning more popular or 'head' types of queries;
2. the volume of an initial query $Q$ is changing during the testing period (we also can detect a drift in an increase of $Q$ volume);
3. search abandonment is a frequent behavior on the initial $SERP$ for $Q$ indicating $SERP$ failure, i.e., the required $url_q$ is missing (as presented in Definition 6).

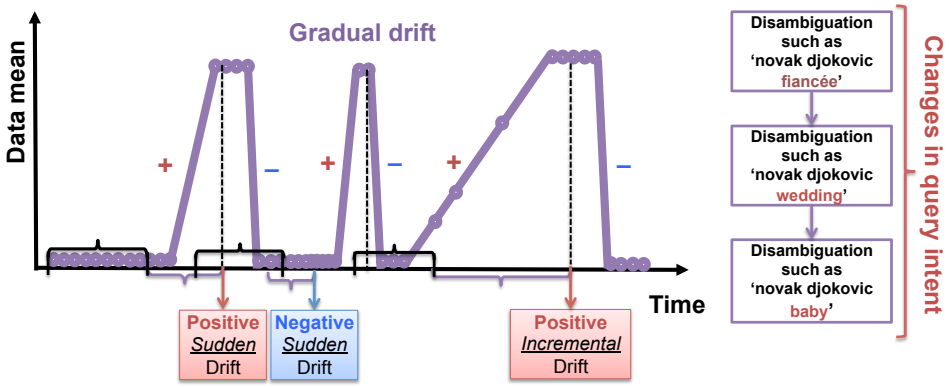



Figure 8.3: A representation of the gradual drift.

### Incremental change

This kind of change is less often studied and can be characterized by a slow change in query intent over a long period of time, as shown on the right hand side of Figure 8.2. An example is the reformulation of the query *'CIKM conference'* to include the specific year or location. This drift is more difficult to detect because it does not necessary require an increasing query volume. However, changes in the fraction of query reformulations [145] can be used to detect incremental drift. Using the secondary metrics the incremental drift can be defined as:

1. the incremental drift is likely concerning less popular or 'tail' types of queries;
2. the volume of initial queries is not changing much during the testing period (we hardly can detect changes in an increase of volume);
3. search abandonment is a frequent behavior of initial $SERPs$ that means the initial $SERP$ has failed. Also in case a required $URL$ is present at the $SERP$ but at a low rank, users tend to reformulate their query rather than explore further results on the $SERP$.

We identify two other types of query intent drift that can be represented as a combination of sudden and incremental types.

### Gradual change

This is a different type of change that is presented in Figure 8.3. It can be viewed as a combination of the sudden and (or) incremental types of changes happening over time. For example, we consider the query *'novak djokovic'* (the famous tennis player) that may change its intents over time. For example, it has a drift in September 2013 on the term *'fiancée'* because the tennis player got engaged. Therefore, $SERP$ for the initial query is missing information about his fiancée and users tend to reformulate because they are interested in this topic.

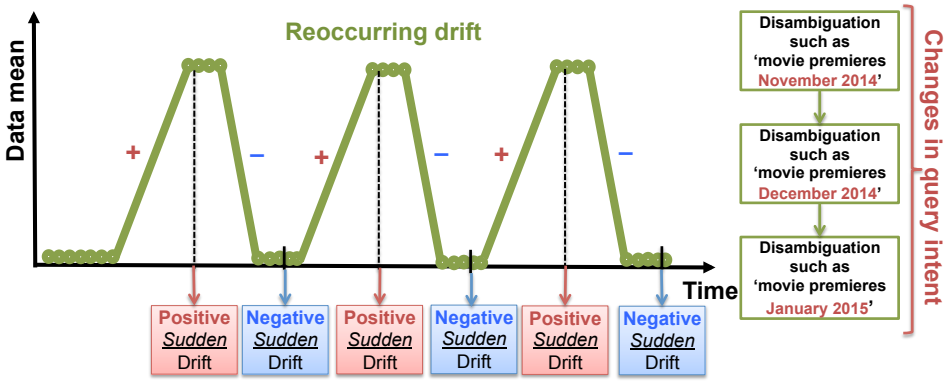



Figure 8.4: A representation of the reoccurring drift.

At some moment in time the couple celebrates their wedding and users' interest changes again. Therefore the news about the engagement of the famous tennis player becomes outdated and users start to reformulate the query *'novak djokovic'* using the reformulation *'wedding'* (if this information is missing from the $SERP$). As a logical continuation of this story the couple has a baby and users are interested to see information about this news, etc. It is important to note that the described changes are happening without any pattern, so some of these drifts may be sudden and some may be incremental.

### Reoccurring change

The final special case of drifts is presented in Figure 8.4. A specific characteristic of the the reoccurring change is that it has a regular type of behavior [209]. The example in Figure 8.4 shows that users reformulate the query *'movies premieres'* regularly according to dates. It is important to note that the reoccurring change is a combination of the same type drifts (sudden or incremental).

It is important to track changes for queries over time in order to understand if the $SERP$ is fixed when it is needed. A *positive* drift is typical for those cases where the reformulation signal is growing over time. Basically, the detected positive drift is a sign of a failing $SERP$ (shown as "+" in Figure 8.3). A *negative* drift is typical for those cases where the probability to reformulate a query is decreasing dramatically (shown as "-" in Figure 8.3). The *negative* drift may be interpreted in the two following ways:

1. the system has reacted to the positive change first and changed the $SERP$. Therefore the number of query revisions dropped down, which means that the problem with user satisfaction has been fixed;

2. the system has not reacted to the positive change in the reformulation signal but the moment has passed and users are no longer interested in a revision.

A detailed algorithm how to identify if a detected drift has positive or negative signs is presented in the next section.

### 8.4.2 Detecting Sudden and Incremental Drifts

Let us first define formally a set of features $\{F_j\}_{j=1}^{4}$ we use to detect changes in user satisfaction with the pair $\langle Q, SERP \rangle$: (1) a reformulation signal ($RS$), (2) a search abandonment signal ($AS$), (3) a query volume signal ($VS$), and (4) an average clicked position signal ($CS$):

- $F_1$: $RS(Q, Q')_{[t_i, t_{i+1}]}$ for the query $Q$ and its reformulation $Q'$ is a probability to reformulate $Q$ to $Q'$ within a particular time period $[t_i, t_{i+1}]$:

$$RS = P\left(Q \rightarrow Q'\right).$$

- $F_2$: $AS(Q)_{[t_i, t_{i+1}]}$ for the query $Q$ is a probability to abandon (to give no clicks) $SERP_Q$ within a particular time period $[t_i, t_{i+1}]$.
- $F_3$: $VS(Q)_{[t_i, t_{i+1}]}$ for the query $Q$ is a frequency of this $Q$ within a particular time period $[t_i, t_{i+1}]$.
- $F_4$: $CS(Q)_{[t_i, t_{i+1}]}$ for the query $Q$ is an average position clicked on $SERP_Q$ within a particular time period $[t_i, t_{i+1}]$.

We call $F_1$ as a primary drift metric of drift and $\{F_j\}_{j=2}^{4}$ as a list of secondary drift metrics. Each of them can be estimated straightforwardly based on observed frequencies in the period: for $RS$, we calculate the probability of reformulation per day, and we use for the period the (observed) average over days ($\mu$).

The proposed algorithm $DTDSAT$ to discover types of changes in user satisfaction is presented in Algorithm 4. It is a straightforward application of the adaptive windowing algorithm from concept drift detection [32], which calculates a theoretically motivated threshold $e$ for observing a significant drift, based on a confidence value $\delta$. We will first explain how $DTDSAT$ works.

**DTDSAT Input.** We assume that we can detect the *sudden drift* within a short period of time ($w_1$) such as from three days up to two weeks. In contrast, the *incremental drift* is detected on a larger time slot ($w_2$) from more than two weeks and up to one month. As the train window ($\Delta t$), we use a fixed period of time for both considered types of drift. We calculate error thresholds: $e_{RS}$, $e_{AS}$, $e_{CS}$, $e_{VS}$ using a standard method described in [32].

**DTDSAT Output.** The algorithm $DTDSAT$ returns an alarm as an output if the drift happens. Additionally, it produces an extra information about a detected drift: a sign: **(1)** the *'positive sign'* means we need to fix $SERP$; **(2)** the *'negative sign'* means users are no longer reformulating $Q$, so no action should be taken. A collected sequence of positive and negative drifts for $Q$ can be used to build a dynamic of changes for $Q$. This dynamic may help to understand if $Q$ has a gradual drift (e. g., Figure 8.3) or a reoccurring one (e. g., Figure 8.4) over some longer period of time (e. g., 6 months or 1 year).

The Algorithm 4 includes the following methods:

**Algorithm 4** Algorithm for Detection the Type of Drift in user SATisfaction ($DTDSAT$). We leave out variables, i.e., $drift$ stands for $drift_{Q,Q',w}$, for readability.

**Require:** the train period $\Delta t = [t_i, t_{i+1}]$;
the test window $w = \{w_1, w_2\}$;
the error thresholds: $e_{RS}$, $e_{AS}$, $e_{CS}$, $e_{VS}$;
**Ensure:** $drift \leftarrow true, false$;
$drift\_positive \leftarrow true, false$;
$serp\_fail \leftarrow true, false$
1: **for** $\{Q, Q'\}_{k=1}^{N}$ **do**
2: $\Delta_{RS} = \mu(RS_{\Delta t+w}(Q, Q')) - \mu(RS_{\Delta t}(Q, Q'))$
3: **if** $|\Delta_{RS}| \geq e_{RS}$ **then**
4: $drift \leftarrow true$
5: **if** $\Delta_{RS} > 0$ **then**
6: $drift\_positive \leftarrow true$
7: $\Delta_{AS} = \mu(AS_{\Delta t+w}(Q)) - \mu(AS_{\Delta t}(Q))$
8: $\Delta_{CS} = \mu(CS_{\Delta t+w}(Q)) - \mu(CS_{\Delta t}(Q))$
9: $\Delta_{VS} = \mu(VS_{\Delta t+w}(Q)) - \mu(VS_{\Delta t}(Q))$
10: **if** ($|\Delta_{AS}| \geq e_{AS}$ **or** $|\Delta_{CS}| \geq e_{CS}$) **then**
11: $serp\_fail \leftarrow true$
12: $url_Q \leftarrow getMissingTopURL()$
13: $driftType \leftarrow getDriftType(|w|, \Delta_{VS})$
14: $fixSerp(url_Q, driftType)$
15: **else**
16: $serp\_fail \leftarrow false$
17: **end if**
18: **else**
19: $drift\_positive \leftarrow false$
20: **end if**
21: **else**
22: $drift \leftarrow false$
23: **end if**
24: **end for**
25: **return** $drift$, $drift\_positive$, $serp\_fail$

- a method $getMissingTopURL()$ returns a missing $url_Q$ as defined in Definition 6. It gets $url_Q$ based on a statistics about the most frequently clicked $URLs$ after issuing drifted reformulation $Q$ to $Q'$.

- a method $getDriftType(\Delta_{AS}, \Delta_{CS}, \Delta_{VS})$ takes into account a statistics about the secondary metrics of drifts in order to estimate a type of drift. There are cases when our system detects a drift on $RS$ but none of the secondary metrics has changed. We called this situation a *'positive reformulation'*. In this case we are not dealing with failed $SERP$.

- a method *fixSerp(*$url_Q$*,driftType)* that will produce a list of $URLs$ that users mostly click on after running query revisions. The top $URLs$ from the list of candidates to be included on the $SERP$ served for the original $Q$, and avoid future user frustration and the need to revise their queries.

Summarizing, in this section, we studied different types of possible query intent drift inspired by the literature on concept drift [79]: sudden, incremental, gradual and reoccurring. We identified relevant parameters, such as the window of change, volume or popularity of queries, and relevant behavioral indicators, such as the probability of reformulation, abandonment rates, and click through rates. For the two main categories of intent drift, we define an unsupervised approach to detect failed SERPs caused by drift. We also showed how the detected changes can be used to improve a ranking of search results.

## 8.5 Experiments and Results

In this section we will study **RQ 8.3** *How effective is our approach on a realistic sample of traffic of a major Internet search engine?*

### 8.5.1 Experimental Data

Our experimental data consists of of massive raw and unfiltered search logs of the commercial search engine `yandex.ru` that were collected during the whole year 2014. Our audience consists of about *25 million users per day.* Our traffic consists of approximately *150 million of queries per day.* In our experimentation, we are dealing with a multilingual traffic that has at least *five* dominant languages.

### 8.5.2 Evaluation Methodology

We now describe our methodology to evaluate the quality of the drifts detection algorithm.

Our algorithm is unsupervised, and detects drift in query intent based on a concept drift technique using a simple, theoretically motivated threshold, needing only a single linear pass through the data. To the best of our knowledge, there is no alternative approach to detect failed SERPs that could function as a baseline. We run our algorithm of Section 8.4 in a simplified form on the logs. First, we choose three fixed windows of 3, 7, and 14 days rather than calculate the optimal window based on the threshold. Second, we use the change in the probability of a revision ($\Delta_{RS}$) as the criterion to select data. Third, we use a single threshold ($e_{RS}$) based on $\delta = 0.1$, with values in the range of $[0.2, 0.5]$, as we did not observe major differences between the settings.

As an approximation, we define the drift type by the size of the test window, so a three days windows size is related to a sudden drift type. However, it is important to note that a fresh intent classifier (based on mostly a query popularity) is already working within the search engine. Let us call it $FIC$. Therefore, most

popular changes in query intents might picked up by $FIC$ and $SERP$ is already fixed for *'head queries'*. As was shown in [164] this can be done within a very short period of time.

For evaluation, we selected *randomly* about 150 examples from different batches. Therefore, in total, we selected 450 examples of drifts for a test set which we use to report the final results. Each detected drift in our test set was evaluated by three judges and we report overall scores. As an evaluation metrics we use accuracy rates. In the current settings we are more interested to obtain rather precise results.

In order to evaluate the obtained results we set up the following evaluation task. Every judge is supplied with the definition of sudden and incremental types of drift. We gave to the judges the following explanation for the drift labels:

1. Drift is detected: *'real' drift* in users intent i.e. new target intent replaces the old target intent: e. g. $Q$ = *'referendum in Crimea'* has a drift on its revision $Q'$ = *'referendum in Crimea 16 march'* in March 2014 (due to some events happening in the world); similarly $Q$ = *'Sochi'* has a drift on its revision $Q'$ =*'Sochi **2014**'* in February 2014 (due to Olympics games that took place in the city Sochi in February 2014);

2. Drift is detected: *'new drift'* in users intent i.e. an another/new target intent added to a multifaceted query: e. g. $Q$ = *'Happy New Year wishes'* has a drift on its revision $Q'$ = *'Happy New Year wishes **2015**'* in December 2014 (due to the fact that people are trying to find the next year); similarly $Q$ = *'RoboCop'* has a drift on its revision $Q'$ = *'RoboCop **2014**'* in February 2014 (due to the release of the new movie);

3. Positive reformulation is detected: it signals about a shift in users intent but we suppose that $SERP$ is not broken in this case: e. g. $Q$ = *'schedule of matches of the World Cup 2014'* has a detected positive reformulation $Q'$ = *'schedule of matches of the World Cup 2014 **on tv**'* in June 2014; similarly $Q$ = *'voice 22.11.2014'* has a detected positive reformulation $Q'$ = *'voice 22.11.2014 **watch online**'* ('Voice' is the popular TV show);

4. It is not a drift;

5. There is not enough information to judge.

In order to evaluate the quality of our procedure to fix failed $SERPs$ we asked judges to check the suggested @$url$ and answer the question *'Would this* @$url$ *be useful on the initial SERP for the query* @$Q$ *at the particular moment in time* @$\Delta t$*?'*. Judges were supplied with the following set of labels:

1. *Yes*, it would be useful to insert the @$url$ to the $SERP$ for the @$Q$ during the time period @$\Delta t$;

2. *No*, it does not make sense to insert the @$url$ to the $SERP$ for the @$Q$ during the time period @$\Delta t$;

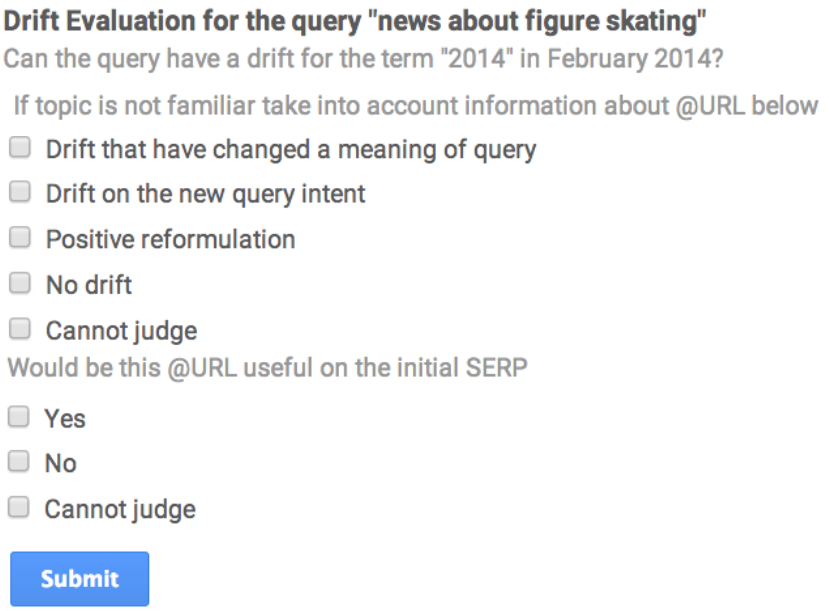



Figure 8.5: A fragment of an evaluation task for the annotators. We suggest the most clicked *url* after a query revision.

3. There is not enough information to judge.

The interface for our labeling procedure is presented in Figure 8.5. As it turned out, judges were struggling to distinguish between the first two categories of Figure 8.5, hence we decide to collapse these two categories into a single case of drift due to a failed $SERP$, consistent with the algorithm in Section 8.4.

### 8.5.3 Experimental Results

We detect 100s of thousands of revisions over a whole year, and over 200,000 unique $\langle Q, Q' \rangle$ pairs. This is a considerable number, but of course still a small fraction of the overall traffic. The set of revision terms is rather varied, with a revision term occurring in 3-4 unique pairs of $\langle Q, Q' \rangle$. Familiar patterns like 'year' revisions (i.e., '2014' or '2015') account just a around 2-3 % of the revisions, and around 17-18 % contains any number. This suggests we capture a wide variety of revisions, beyond those that could be detected based on rules and templates. For the non-numerical revisions, we also see queries and revisions in many languages, show-casing the general applicability of the unsupervised approach.

Table 8.2: Accuracy of drift detection (including positive reformulations)

| | Window | Drift_Accuracy | URL_Accuracy |
|---|---|---|---|
| 1 | 3 days | 0.58 | 0.87 |
| 2 | 7 days | 0.66 | 0.97 |
| 3 | 14 days | 0.91 | 0.91 |
| 4 | Combined | 0.72 | 0.92 |

Table 8.3: Accuracy of failed SERP and positive reformulation detection

| | Window | Drift (Failure) | Positive Reformulation |
|---|---|---|---|
| 1 | 3 days | 0.17 | 0.41 |
| 2 | 7 days | 0.41 | 0.20 |
| 3 | 14 days | 0.80 | 0.11 |
| 4 | Combined | 0.47 | 0.25 |

Table 8.2 shows the results of the drift detection approach for three test windows with 3 days, 7 days, and 14 days, where we look at all cases of drift (failed $SERP$ and positive reformulations) versus the 'no drift' judgment, and on the judgment on the utility to include the URL clicked after revisions on the $SERP$ of the original query. We observe a 72% accuracy for the drift detection and a 92% accuracy for the usefulness of the URL to be included on the original page's $SERP$. While the detection doesn't work flawlessly, these accuracies are a clear indicator of the value of the approach to detect failed $SERPs$. To put this performance into perspective, these number are based on the simplified algorithm based on the probability of query revisions and the theoretical threshold, rather than optimized tuning. The high levels of accuracy for the picked up $URLs$ confirm that these are of interest to be included on the $SERP$ of the original query.

Looking at the breakdown over the duration of the test windows, we see a considerable increase in accuracy for the longer test periods, reaching up to 91% accuracy for the 14 day window. This leads to two observations. First, the approach seems to benefit from more observations to make more reliable judgments, and a revision pattern observed over two weeks is obviously a clearer signal. This also suggests the value of our approach for the detection of incremental change. Second, we expected to obtain high accuracy for popular or head queries in the 3 day window, but observed mostly queries with 1-3 revisions per day. A plausible explanation is that these are picked up and corrected by the $SERP$ already, as recency ranking tools are employed in the search engine that can respond within hours. A possible resolution is to use smaller and adaptive windows, as is done in the concept drift literature, or defining window size in terms of an absolute minimum number of revisions leading to very short window sizes for popular queries.

In Table 8.3 we break down the two cases of drift: due to a genuine shift in intent towards a new direction, hence indicating a failed SERP, or due to

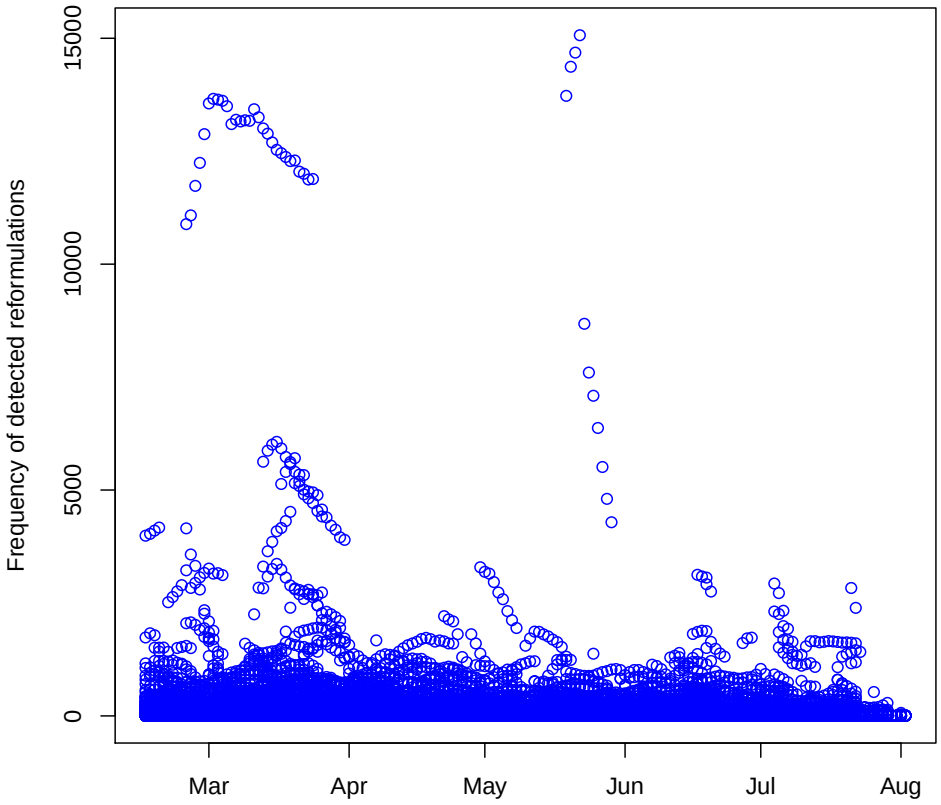



Figure 8.6: Frequency of detected reformulations over time.

the exploration of another aspect or facet of the original request beyond what's presented on the initial SERP. We observe 47% accuracy for the failed SERP detection relative to all detected $\langle Q, SERP \rangle$ pairs, hence the remaining 25% are positive reformulations. In particular in these 47% of the cases it would be important to consider, including the URL clicked after revision into the $SERP$ of the original query. As observed above, the detection accuracy goes significantly up over the larger duration of the detection windows. Over 14 days, no less than 80% of the detected cases indicate a failed $SERP$ due to query intent drift. As observed above, our approach is very effective to detect cases of incremental drift, but less effective to pick up sudden drift over the shortest period.

In this section, we limited ourselves by varying time windows of detection due to patterns of sudden and incremental drifts. Figure 8.6 shows the frequencies of detected query reformulations over time. What we observe is that we detect the same drifts on consecutive days, but also that the revisions may disappear after a period of time. Anecdotal evidence suggests that this can be both due to another drift in query intent, for example for revisions specific to events or months of the year, or due to updates of the SERP served for the original query. This supports

to importance of detecting both positive and negative drift patterns, and also to look at gradual and reoccurring drifts.

Summarizing, in this section, we ran a simplified version of our algorithm on a massive transaction log, and detected over 200,000 pairs of $\langle Q, SERP \rangle$ suspected of failing due to drifting query intents. We observed a reasonable accuracy of drift detection (72%) and a high accuracy of candidate URLs to be included on the $SERP$ of the original query. For incremental change over the longer detection period of 14 days, we detected failed $SERPs$ due to query intent drift with an 80% accuracy. Under the specific conditions of the recency optimized search engine, the performance for detecting sudden change over shorter periods was less effective.

## 8.6 Conclusions

This chapter investigated how the dynamic nature of web content and user intents have consequences for the SERP to be displayed for a particular query. There remain cases of failure where the organic search results on the search engine result page ($SERP$) are outdated, and no relevant result is displayed. This can be caused by temporal query intent drift, where the desired pages for a query are changing over time, and the historical transaction logs privilege the outdated results. Our main research question was: *By analyzing behavioral dynamics at the SERP level, can we detect an important class of detrimental cases (such as search failure) based on changes in observable behavior caused by low user satisfaction?*

We presented an overview of prior work on topic and concept drift, behavioral dynamics, and user satisfaction on the web, with a special focus on the $SERP$ level. We conducted a conceptual analysis of success and failure at the SERP level in order to answer our first research question: **RQ 8.1** *How to include the SERP into the conceptual model of behavioral dynamics on the web? How to identify (un)successful SERPs in terms of drastic changes in observable user behavior?* Specifically, we introduced the concept of a successful and failed SERP and analyzed their behavioral consequences identifying indicators of success and failure. By analyzing success and failure in light of changing query intents over time, we identified an important case of SERP failure due to query intent drift. This suggested an approach to detect a failed SERP due to query intent drift by significant changes in behavioral indicators of failure.

We continued our analysis of different types of drifts in query intent over time, answering our second research question: **RQ 8.2** *Can we distinguish different types of SERP failure due to query intent drift (e. g., sudden, incremental), and when and how should we update the SERP to reflect these changes?* Inspired by the literature on concept drift [79], we studied different changes in query intent: sudden, incremental, gradual and reoccurring, and identified relevant parameters, such as the window of change, volume or popularity of queries, and relevant behavioral indicators, such as the probability of reformulation, abandonment rates, and click through rates. For the two main categories of intent drift, we defined

an unsupervised approach to detect failed SERPs caused by drift, requiring only a single pass through a transaction log.

Finally, we ran experiments on massive raw search logs, answering our third research question: **RQ 8.3** *How effective is our approach on a realistic sample of traffic of a major Internet search engine?* We ran a simplified version of our algorithm and detected over 200,000 pairs of $\langle Q, SERP \rangle$ suspected of failing due to drifting query intents, observing a reasonable accuracy of drift detection (72%) and a high accuracy of candidate URLs to be included on the SERP of the original query. For incremental change over the longer detection period of 14 days, we detected failed SERPs due to query intent drift with an 80% accuracy but, under the specific conditions of the recency optimized search engine, the performance for detecting sudden change over shorter periods was less effective.

As future work, we are further developing the conceptual model, and are running further offline experiments exploring further window sizes, and further features of user dissatisfaction. We are also planning to do online evaluation of how the discovered drifts are useful for fixing SERPs. In addition to the unsupervised methods of this chapter, we are also experimenting with tuning the optimal parameters and threshold based on behavioral features and initial results suggest further improvements.

Real data is messy and has many intricate dependencies, such as continually changing ranking, personalization, customization and localization, and specific tools to update the ranker fast on other signals (i. e., recency ranking). This makes data-driven research a difficult enterprise, and we strongly feel that this should be coupled with theoretical and conceptual analysis. We made a first attempt at this in the current work, where we conduct conceptual analysis to clarify the meaning of core concepts and their relations and dependencies. And as a conceptual model, work with an idealized model that abstracts away from other factors outside the scope of our interest. For example, we observed in the experimental data relatively few or popular queries as those are tackled within hours by recency ranking methods. We viewed the experimental part more as initial validation experiments, mostly used to inform the conceptual model as well as identify the most useful features in the context of real world traffic. For this reason we did not "optimize" for the data using supervised methods, but collected a single set of data for three time windows and analyzed this to assess the value of the variables in the conceptual model, and to further develop our model. We strongly belief that conceptual and experimental research should go hand in hand, and without denying the value of "things that work in practice" we should put equal value on experiments that contribute to our conceptual or theoretical understanding.

# 9 Conclusions

This chapter concludes this dissertation by revisiting research questions from Chapter 1 and discussing our main findings (Section 9.1), and sketching directions for future research (Section 9.2). We focus on the main findings and general lessons, additional detailed findings are in the the conclusion sections of the individual chapters.

## 9.1 Main Findings

The work included in this dissertation emerged from the observation that user searching and browsing behavior online is inherently sensitive to context. We explored the main research question: *How to discover, model and utilize contextual information in order to understand and improve users' searching and browsing behavior on the web?* In a series of empirical studies, we investigated a number of concrete research questions, which we will discuss in turn.

### 9.1.1 Useful Contextual Information

Our first research question was:

> **RQ 1**: *What are the general characteristics of useful contextual information?*

To answer this research question, we introduced a formal definition of useful context. We formulated the context discovery as an optimization problem and called it the *contextual principle*. We provided intuitive proofs showing that that the problem of finding the best model can be solved by considering the subproblems of finding optimal contextual models. This result provided us with theoretical judgment for customization and exploitation of contextual information in predictive analytics.

Further, the contextual principle is directly used in Chapters 2 and 3. In Chapters 4, 5, and 6 while experimenting with intelligent assistants, we did not

use the contextual principle directly but assumed that introducing additional contextual information by expanding the feature set, e. g. user interaction signals, speech recognition quality etc, would improve the prediction of user satisfaction. In Chapters 7 and 8 we applied the contextual principle in monitoring settings, focusing on cases where a change in user intents is happening due to some implicit context (e. g. news event) resulting in a decrease of user satisfaction with the SERP.

Our main finding for **RQ 1** is that we can characterize useful contextual information in an abstract, formal way, within a typical machine learning or optimization problem underlying most web applications.

### 9.1.2 Explicit Contextual Information

Our second research question was:

> **RQ 2**: *How to identify useful contextual information from the available list of explicit contexts?*

We started our investigation of this question in Chapter 2, where our main goal was to predict the next user action during his website browsing [33, 206, 256]. Predictive methods help to infer user preferences in order to understand how we can engage users. We built contextual Markov models based on the StudyPortals using the geographical location as explicit contextual information.

However, according to the contextual principle of Chapter 2, our experiments showed that geographical location has no useful contextual information in this setting. The set of local models trained separately for each geographical location was not outperforming the global model trained based on the whole log. A possible explanation is that the general audience of StudyPortals consists of students who have a certain level of education, approximately same age and having high proficiency in English.

Our finding is that location is not a universally useful contextual information, and adding context is no panacea making it non-trivial to determine upfront what the potential value of contextual information for a given web application.

We continued our investigation of **RQ 2** in Chapter 3, where our main goal was to serve complex exploratory recommendations to satisfy their user's needs. Standard personalization and recommender systems rely on rich user profiles but the majority of users are new or visit highly infrequently—we face a continuous cold start recommendation problem (*CoCoS*). We specifically studied this problem in the context of one of the largest travel websites Booking.com and its Destination Finder service. We introduced and characterized the *CoCoS* that happens when users (*UCoCoS*) or/and items (*ICoCoS*) remain 'cold' for a long time, and can even 'cool down' again after some time due to some external contextual signals. Since users visit infrequently and have volatile interests, we cannot rely on historical user interactions. In this setting, mining situational profiles to which we can map an incoming user is an effective way to deal with data sparsity and changing user interests.

We presented an approach for discovering and using contextual user profiles that uses the contextual principle. We demonstrated based on offline data that users in different contextual profiles exhibited different behavior. We used the user agent data and time related information to discover generic contextual profiles. By setting up an online A/B testing evaluation, we compared our contextual travel recommendations to a non-contextual ranker corresponding to the current live system. We observed an increase in user engagement, with higher click-through rates (20%) and higher clicks per user (21%).

Our finding is that our contextual ranking approach showed a dramatic increase in user engagement over a non-contextual baseline, clearly demonstrating the value of contextualized profiles in a real world application that suffers from *CoCoS*.

Our main finding for **RQ 2** is that applying the contextual principle to discover useful contextual information from standard features available in every web transaction logs is a straightforward approach that can have a clear impact on ranking quality in practice.

### 9.1.3 Implicit Contextual Information

Our third research question was:

> **RQ 3**: *How to discover users' behavioral aspects as contextual information?*

We started our investigation of this question in Chapter 2, where we were looking for ways to improve next action prediction while users are browsing a web site. The users' historical data was summarized as the user navigation graph. We discovered two groups of nodes in the navigation graph that reflect different types of user behavior using the community detection technique [35]: (1) an expert user, who is experienced with the website interface or searches extensively to find required information; and (2) a novice user, who needs more time to learn about a website or is not interested much in its content. Applying the contextual principle, we discovered changes in user intents that are happening during a single web session. The discovered implicit contexts dramatically improved the prediction accuracy of user trails.

Our finding is that if we can identify implicit useful contexts then the local Markov models outperform the single global Markov model, and that even if context is not useful, the local models will still perform as good as the global model.

We continued with **RQ 3** in Chapter 4 by investigating key behavioral aspects that determine user satisfaction for different scenarios of intelligent assistant usage. We proposed three main types of scenarios of use: (1) controlling the device; (2) searching mobile web; and (3) search dialogues. The scenarios were identified on the basis of two main factors: (1) their proportional existence in the logs of a commercial intelligent assistant; and (2) the way requests are handled at the intelligent assistant backend (e. g. user requests are redirected to the different

services and they serve different interfaces). We designed a series of user studies tailored to the three scenarios of intelligent assistants use, with questionnaires on variables potentially affecting to user satisfaction. The tasks used in the experiment were based on an extensive analysis of logs of a commercial intelligent assistant. We collected participant's responses on their satisfaction with the task, their ability to complete a task, and the estimated effort it took. We found that effort is a key component of user satisfaction across the different intelligent assistants scenarios. We demonstrated the presence of 'good abandonment' in the web search scenario, and concluded that to measure user satisfaction we need to investigate the other forms of interaction signals beyond clicks or reformulations as used in desktop search. We looked at user satisfaction as 'a user journey towards an information goal where each step is important,' and showed the importance of session context on user satisfaction. Our experimental results showed that user satisfaction cannot be measured by averaging over satisfaction with sub-tasks. Hence, frustration with some steps in a user's 'journey' can greatly affect their overall satisfaction.

Our finding is that the factors contributing to overall satisfaction with a task are different between the scenarios: for the device control scenario, task completion is highly correlated with user satisfaction—it either worked or it did not; for information seeking scenarios, user satisfaction is more related to effort than task completion; and for information seeking scenarios, we found that task-level satisfaction cannot be reduced to query or impression-level satisfaction.

We continued the investigation of **RQ 3** by analyzing prediction of user satisfaction with search dialogues taking into account contextual information such as touch gestures and voice interaction. We defined user satisfaction with search dialogues in the generalized form, which showed understanding the nature of user satisfaction as an aggregation of satisfaction with all dialogue's tasks and not as a satisfaction with all dialogue's queries separately. To predict user satisfaction, we used the following kinds of interactions: clicks (or 'taps' in terms of touches on mobile platforms), other touch interactions and voice features. The baseline was predicting user satisfaction using clicks and queries features. By conducting experiments we proved that features derived from voice and especially from touch interactions add significant gain in accuracy over the baseline. To understand how to efficiently select features depending on different types of queries, we proposed three techniques: using only features of queries resulting in structured interface; calculating a single set of features for queries resulting in structured interface and queries resulting in general SERP; and calculating own set of features for each group of queries resulting in structured interface and queries resulting in general SERP. We conducted analysis and showed that the third technique is the most accurate to model user satisfaction. This technique improves accuracy from 71% to 81% over the baseline. Additionally, we analyzed the impact of each class of interaction features, aiding to our understanding of the causes of (dis)satisfaction. This showed that users expect to find answers on the SERP directly without putting in any 'additional effort' (e. g. scrolling). Our analysis showed a strong negative correlation between user satisfaction and swipe actions. Additionally, we

demonstrated that users are not satisfied if the intelligent assistant cannot answer their query explicitly and redirects them to a general mobile SERP.

We continued our study of **RQ 3** in Chapter 6 by investigating if gesture features are useful for differentiating between good and bad abandonment in mobile search. By formulating a supervised classification experiment, we showed how user gesture features perform significantly better than query and session features as well as other click-based baselines. We showed this on a high quality dataset collected through a user study (discussed above) and verify the results on a crowdsourced dataset. We also conducted an A/B experiment, whereby our good abandonment aware model was able to detect a significant decrease in our metric. Through a correlation analysis, we showed how time spent interacting with answers on a SERP are positively correlated with satisfaction and good abandonment. By contrast, swipe interactions and time spent interacting with organic search results were negatively correlated with satisfaction. By analyzing data collected through our user study, we showed how good abandonment can be driven by many elements on a SERP, such as answers (and even different types of answer), snippets and images and conclude that good abandonment is due to many factors.

Our finding is that touch based features dramatically improve the prediction quality of user satisfaction with search dialogue and the detection of good abandonment for mobile web search.

Our main finding for **RQ 3** is that introducing users' behavioral aspects as contextual information is beneficial to improve user experience in various web applications, and that interaction behavior even at the micro-level holds important contextual cues.

### 9.1.4 Dynamic Contextual Information

Our fourth research question was:

> **RQ 4**: *How to define and to detect changes in user satisfaction with retrieved search results?*

We investigated how the dynamic nature of web content and user intents have consequences for the search engine result page (SERP) to be displayed for a particular query in Chapters 7 and 8. In particular, there remain cases of failure where the organic search results SERP are outdated, and no relevant result is displayed. This can be caused by temporal query intent drift, where the desired pages for a query are changing over time, and the historical transaction logs privilege the outdated results. We presented an overview of prior work on topic and concept drift, behavioral dynamics, and user satisfaction on the web, with a special focus on the SERP-level. We conducted a conceptual analysis of how to include the SERP into the conceptual model of behavioral dynamics on the web. Specifically, we introduced the concept of a successful and failed SERP and analyzed their behavioral consequences identifying indicators of success and failure. By analyzing success and failure in light of changing query intents over

time, we identified an important case of SERP failure due to query intent drift. This suggested an approach to detect a failed SERP due to query intent drift by significant changes in behavioral indicators of failure. Our general conclusion is that more detrimental cases in terms of user satisfaction lead to larger changes in observable user behavior and hence more handles to detect them.

We continued our analysis of different types of drifts in query intent over time that consists of two parts: First, how to distinguish different types of SERP failure due to query intent drift. Second, when and how should we update the SERP to reflect these changes. Inspired by the literature on concept drift [79], we studied different changes in query intent: sudden, incremental, gradual and reoccurring, and identified relevant parameters, such as the window of change, volume or popularity of queries, and relevant behavioral indicators, such as the probability of reformulation, abandonment rates, and click through rates. For the two main categories of intent drift, we define an unsupervised approach to detect failed SERPs caused by drift, requiring only a single pass through a transaction log. We also showed how the detected changes can be used to improve a ranking of search results.

To evaluate our methods, we ran experiments on massive raw search logs from Microsoft Bing (USA traffic) and Yandex (Russian traffic). Our extensive evaluations showed that the proposed methods demonstrated high accuracy to detect changes in user satisfaction with the pairs $\langle Q, SERP \rangle$. We also concluded that our methods are language independent.

Our main finding for **RQ 4** is that real data is messy and has many intricate dependencies, such as continually changing ranking, personalization, customization and localization, and specific tools to update the ranker fast on other signals (i. e. recency ranking). This makes data-driven research a difficult enterprise, and we strongly feel that this should be coupled with theoretical and conceptual analysis. We made a first attempt at this, where we conduct conceptual analysis to clarify the meaning of core concepts and their relations and dependencies. And as a conceptual model, work with an idealized model that abstracts away from other factors outside the scope of our interest. For example, we observed in the experimental data relatively few or popular queries as those are tackled within hours by recency ranking methods.

## 9.2 Future Work

This dissertation resulted in insights and techniques for enabling contextual information to improve searching and browsing behavior. Beyond these, it opens up many interesting and important directions for future work. Below, we outline three main areas: rich situational context on mobile devices (Section 9.2.1), meaningful interpretations of rich interaction features (Section 9.2.2), and exploratory contextual suggestions (Section 9.2.3).

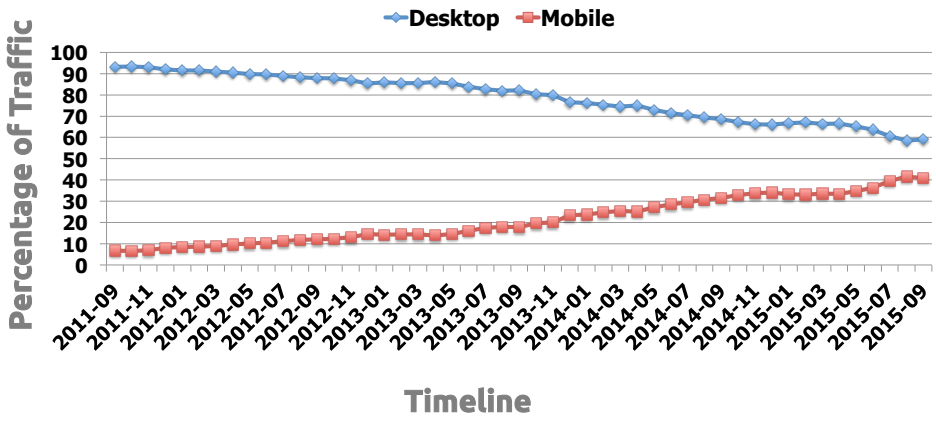



Figure 9.1: The usage of mobile on the web versus desktop. The presented statistics is taken from [220].

### 9.2.1 Situational Contextual on Mobile Devices

Recent years have witnessed a rapid explosion in the usage of mobile devices on the web. According to recent surveys, web browsing on mobile devices increased from 8% in September 2011 till 40% in September 2015 as presented in Figure 9.1. According to the largest search engine, mobile search surpassed desktop search in 2015 [82]. Searching and browsing behavior on mobile devices is different than on desktop for several reasons, but our understanding of these differences is still fragmented at best.

An obvious difference of user behavior on mobile devices are indeed "mobile" and used at various locations. Therefore, we are dealing with much a richer space of potential user situation compared to the relatively static desktop environment, e. g. while driving, in the bus, on the way, a slow connection etc. These conditions can have a great impact on user satisfaction with intelligent assistants. Similar experiences can be satisfying in one situation, e. g. a user needing to find a closest gas station while sitting in a hotel lobby with a fast wi-fi connection and the intelligent assistant is able to understand his request only the second time, and it can be totally frustrating in another situation, e. g. when the same user is driving and running out of gas.

In future work, we could study meaning situational context in far greater detail, allowing us to reason about how a user's current environment impacts his satisfaction. Moreover, answering **RQ 4** we showed that user behavior is inherently sensitive to changes in the environment. This impacts the mobile search even more due to the constantly changing environment.

### 9.2.2 Interpreting Multimodal Interaction Logs

Unlike traditional desktop computers with large displays and mouse-keyboard interactions, touch enabled mobile devices have small displays and offer a variety of touch interactions, including swiping and zooming. A new generation of intelligent assistants, powered by voice, such as Apple's Siri, Microsoft's Cortana, Google Now, etc. have become a common feature on mobile devices. It has resulted in a new way of interacting with search tools which we called search dialogues. In this mode, a conversation takes place between the user and the intelligent assistant: the user speaks to the intelligent assistant, it responds and the user speaks back, frequently referring to the subject of the previous request. This conversational search is a more natural way for people to communicate and is often faster and more convenient (e. g. while driving) than typing. In this dissertation we have started an exploration of this new search interface by using interaction signals to infer user satisfaction. Answering our **RQ 3** we have shown that users' behavioral aspects are extremely important to predict user satisfaction with intelligent assistants.

In future work, we can infer many potentially meaningful contextual cues from the fine grained interaction data. For example, the speech signal has information on gender, first versus second language speaker, indoors or out doors, etc., and the touch signals may tell whether our user is left- or right-handed, revealing personality characteristics, the position of the user's hands on the device, etc. The increasing number of sensors in mobile devices present a sheer endless number of new opportunities.

### 9.2.3 Exploratory Contextual Suggestions

Recent years have witnessed the emergence of various proactive systems such as Google Now and Microsoft Microsoft Cortana. In these systems, relevant content is presented to users based on their context without a query. Similar ideas featured prominently on a recent IR research agenda [16]. Interestingly, despite the increasing popularity of such services, there is very little known about how users interact with them. In the literature it is called information cards, and it was demonstrated that the usage patterns of these cards vary depending on time and location [211].

Current information cards are static and focus on exact answers, yet more complex information needs require a more dynamic and exploratory results. An example of such system be able to respond to queries like *What should I do tonight?* serving dynamic information cards where suggestion are based on:

- where is a user? what are his current preferences and mood?
- what is happening around a user that would suit a user at this moment of time?

Two important open problems are when to proactively surface such recommendations, and how to facilitate interactive search over such dynamic and personalized information cards.

In future work, we can develop systems that integrate information cards into a conversational search interface, surfacing dynamic information cards without the need to enter a query, but making the browsing behavior over dynamic information cards a way to ellicit an underlying complex situational query.

# Bibliography


[1] G. D. Abowd, A. K. Dey, P. J. Brown, N. Davies, M. Smith, and P. Steggles. Towards a better understanding of context and context-awareness. *HUC*, pages 304–307, 1999.

[2] G. Adomavicius and Y. Kwon. New recommendation techniques for multicriteria rating systems. *EXPERT*, 22(3):48–55, 2007.

[3] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. *TKDE*, 17:734–749, 2005.

[4] G. Adomavicius and A. Tuzhilin. Context-aware recommender systems. *CARS*, 2010.

[5] G. Adomavicius, R. Sankaranarayanan, S. Sen, and A. Tuzhilin. Incorporating contextual information in recommender systems using a multidimensional approach. *ACM Transactions on Information Systems (TOIS)*, 23 (1):103–145, 2005.

[6] G. Adomavicius, N. Manouselis, and Y. Kwon. *Multi-Criteria Recommender Systems*, volume 768-803. Recommender Systems Handbook, Springer, 2011.

[7] D. Agarwal and B.-C. Chen. Regression-based latent factor models. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 19–28, 2009.

[8] D. Agarwal and B.-C. Chen. flda: matrix factorization through latent dirichlet allocation. In *Proceedings of the ACM International conference on Web Search and Data Mining (WSDM)*, pages 91–100, 2010.

[9] M. Ageev, Q. Guo, D. Lagun, and E. Agichtein. Find it if you can: a game for modeling different types of web search success using interaction data. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, 2011.

[10] M. Ageev, D. Lagun, and E. Agichtein. Improving search result summaries by using searcher behavior data. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 13–22, 2013.

[11] E. Agichtein, E. Brill, and S. T. Dumais. Improving web search ranking by incorporating user behavior information. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 19–26, 2006.

[12] E. Agichtein, E. Brill, S. T. Dumais, and R. Ragno. Learning user interaction models for predicting web search result preferences. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 3–10, 2006.

[13] A. Ahmed, Y. Low, M. Aly, V. Josifovski, and A. J. Smola. Scalable distributed inference of dynamic user interests for behavioral targeting. In

*Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 114–122, 2011.

[14] A. Al-Maskari, M. Sanderson, and P. Clough. The relationship between ir effectiveness measures and user satisfaction. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 773–774, 2007.

[15] J. Allan. Incremental relevance feedback for information filtering. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 270–278, 1996.

[16] J. Allan, B. Croft, A. Moffat, and M. Sanderson. Frontiers, challenges, and opportunities for information retrieval: Report from swirl 2012 the second strategic workshop on information retrieval in lorne. *ACM SIGIR Forum*, 46(1):2–32, 2012.

[17] A. O. Alves and F. C.Pereira. Making sense of location context. In *Proceedings of the International Workshop on Context Discovery and Data Mining (ContextDD)*, 2012.

[18] M. Aly, A. Hatch, V. Josifovski, and V. K. Narayanan. Web-scale user modeling for targeting. In *Proceedings of the International Conference on World Wide Web (WWW)*, 2012.

[19] A. Arampatzis and A. van Hameran. The score-distributional threshold optimization for adaptive binary classification tasks. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 285–293, 2001.

[20] E. Bakshy and D. Eckles. Uncertainty in online experiments with dependent data: an evaluation of bootstrap methods. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 1303–1311, 2013.

[21] L. Baltrunas and F. Ricci. Context-based splitting of item ratings in collaborative filtering. In *Proceedings of the ACM Conference on Recommender systems (RecSys)*, pages 245–248, 2009.

[22] L. Baltrunas and F. Ricci. Context-dependent items generation in collaborative filtering. In *CARS*, 2009.

[23] L. Baltrunas and F. Ricci. Experimental evaluation of context-dependent collaborative filtering using item splitting. *User Modeling and User-Adapted Interaction*, 24:7–34, 2014.

[24] T. Bao, H. Cao, E. Chen, J. Tian, and H. Xiong. An unsupervised approach to modeling personalized contexts of mobile users. *Knowl. Inf. Syst.*, 31(2): 345–370, 2012.

[25] Z. Bar-Yossef and N. Kraus. Context-sensitive query auto-completion. In *Proceedings of the international conference on World Wide Web (WWW)*, pages 107–116, 2011. doi: 10.1145/1963405.1963424. URL `http://doi.ACM.org/10.1145/1963405.1963424`.

[26] R. Begleiter, R. El-Yaniv, and G. Yona. On prediction using variable order markov models. *Journal of Artificial Intelligence Research (JAIR)*, 22:385–421, 2004.

[27] F. Belém, R. L. T. Santos, J. M. Almeida, and M. A. Gonçalves. Topic

diversity in tag recommendation. In *Proceedings of the ACM Conference on Recommender systems (RecSys)*, pages 141–148, 2013.

[28] P. N. Bennett, M. Shokouhi, and R. Caruana. Implicit preference labels for learning highly selective personalized rankers. In *Proceedings of the International Conference on The Theory of Information Retrieval (ICTIR)*, pages 291–300, 2015. doi: 10.1145/2808194.2809464. URL `http://doi.ACM.org/10.1145/2808194.2809464`.

[29] L. Bernardi, J. Kamps, J. Kiseleva, and M. J. I. Müller. The continuous cold start problem in e-commerce recommender systems. In *Proceedings of the Workshop on New Trends on Content-Based Recommender Systems co-located with ACM Conference on Recommender Systems*, pages 30–33, 2015.

[30] M. S. Bernstein, J. Teevan, S. Dumais, D. Liebling, and E. Horvitz. Direct answers for search queries in the long tail. In *Proceedings of the ACM conference on Human Factors in Computing Systems (CHI)*, pages 237–246, 2012.

[31] J. Besser, M. Larson, and K. Hofmann. Podcast search: user goals and retrieval technologies. *Online Information Review (OIR)*, 34(3):395–419, 2010.

[32] A. Bifet and R. Gavaldá. Learning from time-changing data with adaptive windowing. In *SIAM International Conference on Data Mining (SDM)*, 2007.

[33] M. Bilenko and R. W. White. Mining the search trails of surfing crowds: Identifying relevant websites from user activity. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 51–60, 2008.

[34] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *JMLR*, 3:993–1022, 2003.

[35] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 10, 2008.

[36] C. Bolchini, C. A. Curino, E. Quintarelli, F. A. Schreiber, and L. Tanca. A data-oriented survey of context models. In *SIGMOD*, 2007.

[37] J. Borges and M. Levene. Evaluating variable-length markov chain models for analysis of user web navigation sessions. *IEEE Trans. Knowl. Data Eng. (TKDE)*, 19(4):441–452, 2007.

[38] P. Borlund. The IIR evaluation model: a framework for evaluation of interactive information retrieval systems. *Inf. Res. (IRES)*, 8(3), 2003.

[39] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[40] P. Brown, J. Bovey, and X. Chen. Context-aware applications: From the laboratory to the marketplace. *IEEE Personal Comm*, 4:58–64, 1997.

[41] D. J. Campbell. Task complexity: A review and analysis. *The Academy of Management Review*, 1(13):40 – 52, 1988.

[42] P. G. Campos, F. Díez, and I. Cantador. Time-aware recommender systems: a comprehensive survey and analysis of existing evaluation protocols. *User Model. User-Adapt. Interact. (UMUAI)*, 24(1-2):67–119, 2014.

[43] H. Cao, D. H. Hu, D. Shen, D. Jiang, J.-T. Sun, E. Chen, and Q. Yang.

Context-aware query classification. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, 2009.

[44] H. Cao, T. Bao, Q. Yang, E. Chen, and J. Tian. An effective approach for mining mobile user habits. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*, pages 1677–1680, 2010.

[45] Census Bureau. E-stats: Measuring the electronic economy. https://www.census.gov/econ/estats/, 2015.

[46] D. Chakrabarti, D. Agarwal, and V. Josifovski. Contextual advertising by combining relevance with click feedback. In *Proceedings of the International Conference on World Wide Web (WWW)*, 2008.

[47] X. Chen and X. Zhang. A popularity-based prediction model for web prefetching. *Computer*, 36(6):63–70, 2003.

[48] Y. Chen, D. Pavlov, and J. F. Canny. Large-scale behavioral targeting. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 209–218, 2009.

[49] Y. Chen, Y. Liu, K. Zhou, M. Wang, M. Zhang, and S. Ma. Does vertical bring more satisfaction? predicting search satisfaction in a heterogeneous environment. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*, pages 1581–1590, 2015.

[50] F. Chierichetti, R. Kumar, P. Raghavan, and T. Sarlós. Are web users really markovian? In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 609–618, 2012.

[51] L. B. Chilton and J. Teevan. Addressing people's information needs directly in a web search result page. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 27–36, 2011.

[52] F. C. T. Chua, R. J. Oentaryo, and E.-P. Lim. Modeling temporal adoptions using dynamic matrix factorization. In *ICDM*, pages 91–100, 2013.

[53] A. Chuklin and P. Serdyukov. How query extensions reflect search result abandonments. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 1087–1088, 2012.

[54] A. Chuklin and P. Serdyukov. Good abandonments in factoid queries. In *WWW (Companion Volume)*, pages 483–484, 2012.

[55] A. Chuklin and P. Serdyukov. Potential good abandonment prediction. In *WWW (Companion Volume)*, pages 485–486, 2012.

[56] N. Craswell, O. Zoeter, M. J. Taylor, and B. Ramsey. An experimental comparison of click position-bias models. In *Proceedings of the ACM International conference on Web Search and Data Mining (WSDM)*, pages 87–94, 2008.

[57] N. Dai, M. Shokouhi, and B. D. Davison. Learning to rank for freshness and relevance. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 95–104, 2011.

[58] A. Dean-Hall, C. L. A. Clarke, J. Kamps, and P. Thomas. Evaluating contextual suggestion. In *Proceedings of the Text REtrieval Conference*

*(TREC)*, 2013.

[59] A. Dean-Hall, C. L. A. Clarke, J. Kamps, P. Thomas, and E. M. Voorhees. Overview of the trec 2014 contextual suggestion track. In *Proceedings of the Text REtrieval Conference (TREC)*, 2014.

[60] A. Dean-Hall, C. L. A. Clarke, J. Kamps, and J. Kiseleva. Online evaluation of point-of-interest recommendation systems. In *Proceedings of SCST@ECIR*, 2015.

[61] A. Dean-Hall, C. L. A. Clarke, J. Kamps, J. Kiseleva, and E. M. Voorhees. Overview of the TREC 2015 contextual suggestion track. In *Proceedings of the Text REtrieval Conference (TREC)*, 2015.

[62] A. Deng, T. Li, and Y. Guo. Statistical inference in two-stage online controlled experiments with treatment selection and validation. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 609–618, 2014.

[63] M. Deshpande and G. Karypis. Selective markov models for predicting web page accesses. *ACM Trans. Internet Techn. (TOIT)*, 4((2)):163–184, 2004.

[64] A. Dey, G. Abowd, and D. Salber. A conceptual framework and a toolkit for supporting the rapid prototyping of contextaware applications. *Human Computer Interaction*, 2:97–166, 2001.

[65] A. Diriye, R. White, G. Buscher, and S. T. Dumais. Leaving so soon?: understanding and predicting web search abandonment rationales. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*, pages 1025–1034, 2012.

[66] A. Dong, Y. Chang, Z. Zheng, G. Mishne, J. Bai, R. Zhang, K. Buchner, and C. L. F. Diaz. Towards recency ranking in web search. In *Proceedings of the ACM International conference on Web Search and Data Mining (WSDM)*, pages 11–20, 2010.

[67] A. Dong, R. Zhang, P. Kolari, J. Bai, F. Diaz, Y. Chang, Z. Zheng, and H. Zha. Time is of the essence: improving recency ranking using twitter data. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 331–340, 2010.

[68] X. Dongshan and S. Junyi. A new markov model for web access prediction. *Computing in Science and Engineering*, 4(6):34–39, 2002.

[69] A. Dries and U. Ruckert. Adaptive concept drift detection. In *SIAM International Conference on Data Mining (SDM)*, pages 233–244, 2009.

[70] A. Drutsa, G. Gusev, and P. Serdyukov. Engagement periodicity in search engine usage: Analysis and its application to search quality evaluation. In *Proceedings of the ACM International conference on Web Search and Data Mining (WSDM)*, pages 27–36, 2015.

[71] A. Drutsa, G. Gusev, and P. Serdyukov. Future user engagement prediction and its application to improve the sensitivity of online experiments. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 256–266, 2015.

[72] M. Duggan and A. Smith. Cell Internet Use 2013, 2013. URL `http://www.pewinternet.org/2013/09/16/cell-internet-use-2013/`.

[73] G. Dupret and M. Lalmas. Absence time and user engagement: evaluating

ranking functions. In *Proceedings of the ACM International conference on Web Search and Data Mining (WSDM)*, pages 173–182, 2013.

[74] W. Fan. On the optimality of probability estimation by random decision trees. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 336–341, 2004.

[75] H. A. Feild, J. Allan, and R. Jones. Predicting searcher frustration. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 34–41, 2010.

[76] J. L. Fleiss. Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76(5):378–382, 1971.

[77] S. Fox, K. Karnawat, M. Mydland, S. T. Dumais, and T. White. Evaluating implicit measures to improve web search. *ACM Transactions on Information Systems (TOIS)*, 23(2):147–168, 2005.

[78] J. H. Friedman. Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29(5):1189–1232, 2001.

[79] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia. A survey on concept drift adaptation. *ACM Computing Surveys*, 46(4):44:1–44:37, 2014.

[80] J. Gama, I. Zliobaite, A. Bifet, M. Pechenizkiy, and A. Bouchachia. A survey on concept drift adaptation. *ACM Computing Surveys*, 46(4), 2014.

[81] Google Inc. OMG! mobile voice survey reveals teens love to talk. Official Blog, 2014. `https://googleblog.blogspot.nl/2014/10/omg-mobile-voice-survey-reveals-teens.html`.

[82] Google Inc. Building for the next moment. Inside Adwords, 2015. `http://adwords.blogspot.nl/2015/05/building-for-next-moment.html`.

[83] M. Grbovic, V. Radosavljevic, N. Djuric, N. Bhamidipati, and A. Nagarajan. Gender and interest targeting for sponsored post advertising at tumblr. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 1819–1828, 2015.

[84] Q. Guo and E. Agichtein. Ready to buy or just browsing? detecting web searcher goals from interaction data. In *Proceeding of the International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 130–137, 2010.

[85] Q. Guo and E. Agichtein. Beyond dwell time: Estimating document relevance from cursor movements and other post-click searcher behavior. In *Proceedings of the International conference on World Wide Web (WWW)*, pages 569–578, New York, New York, USA, 2012. ACM Press.

[86] Q. Guo, R. W. White, Y. Zhang, B. Anderson, and S. T. Dumais. Why searchers switch: understanding and predicting engine switching rationales. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 335–344, 2011.

[87] Q. Guo, S. Yuan, and E. Agichtein. Detecting success in mobile search from interaction. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 1229–1230, 2011.

[88] Q. Guo, D. Lagun, and E. Agichtein. Predicting web search success with fine-grained interaction data. In *Proceedings of the ACM International Con-*

*ference on Information and Knowledge Management (CIKM)*, pages 2050–2054, 2012.

[89] Q. Guo, H. Jin, D. Lagun, S. Yuan, and E. Agichtein. Towards estimating web search result relevance from touch interactions on mobile devices. In *CHI Extended Abstracts 2013*, pages 1821–1826, 2013.

[90] Q. Guo, H. Jin, D. Lagun, S. Yuan, and E. Agichtein. Mining touch interaction data on mobile devices to predict web search result relevance. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 153–162, 2013.

[91] N. Hariri, B. Mobasher, and R. D. Burke. Context-aware music recommendation based on latent topic sequential patterns. In *Proceedings of the ACM Conference on Recommender systems (RecSys)*, pages 131–138, 2012.

[92] N. Hariri, B. Mobasher, and R. D. Burke. Query-driven context aware recommendation. In *Proceedings of the ACM Conference on Recommender systems (RecSys)*, pages 9–16, 2013.

[93] H. Hashemi, C. L. A. Clarke, A. Dean-Hall, J. Kamps, and J. Kiseleva. On the reusability of open test collections. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 827–830, 2015.

[94] S. H. Hashemi, C. L. A. Clarke, A. Dean-Hall, J. Kamps, and J. Kiseleva. An easter egg hunting approach to test collection building in dynamic domains. 2016. EVIA@NTCIR.

[95] A. Hassan. A semi-supervised approach to modeling web search satisfaction. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 275–284, 2012.

[96] A. Hassan and R. W. White. Personalized models of search satisfaction. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*, pages 2009–2018, 2013.

[97] A. Hassan, R. Jones, and K. L. Klinkner. Beyond DCG: User behavior as a predictor of a successful search. In *Proceedings of the ACM International conference on Web Search and Data Mining (WSDM)*, pages 221–230, 2010.

[98] A. Hassan, X. Shi, N. Craswell, and B. Ramsey. Beyond clicks: query reformulation as a predictor of search satisfaction. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*, pages 2019–2028, 2013.

[99] A. Hassan Awadallah, R. W. White, S. T. Dumais, and Y.-M. Wang. Struggling or exploring?: disambiguating long search sessions. In *Proceedings of the ACM International conference on Web Search and Data Mining (WSDM)*, pages 53–62, 2014.

[100] A. Hawalah and M. Fasli. Utilizing contextual ontological user profiles for personalized recommendations. *Expert Syst. Appl. (ESWA)*, 41(10):4777–4797, 2014.

[101] L. P. Heck, D. Hakkani-Tür, M. Chinthakunta, G. Tür, R. Iyer, P. Parthasarathy, L. Stifelman, E. Shriberg, and A. Fidler. Multi-modal conversational search and browse. In *SLAM@INTERSPEECH*, pages 96–101, 2013.

[102] K. Henricksen and J. Indulska. Modelling and using imperfect context information. In *Proceedings of the PerCom Workshops*, pages 33–37, 2004.

[103] K. Henricksen and J. Indulska. Developing context-aware pervasive computing applications: Models and approach. *Journal of Pervasive and Mobile Computing*, 2(1):37–64, 2006.

[104] S. Hido, T. Idé, H. Kashima, H. Kubo, and H. Matsuzawa. Unsupervised change analysis using supervised learning. In *PAKDD*, pages 148–159, 2008.

[105] Y.-C. Ho, Y.-T. Chiang, and J. Hsu Yung-Jen. Who likes it more?: mining worth-recommending items from long tails by modeling relative preference. In *Proceedings of the ACM International conference on Web Search and Data Mining (WSDM)*, pages 253–262, 2014.

[106] D. J. Hu, R. Hall, and J. Attenberg. Style in the long tail: Discovering unique interests with latent variable models in large scale social e-commerce. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 1640–1649, New York, NY, USA, 2014. ACM.

[107] J. Huang and A. Diriye. Web user interaction mining from touch enabled mobile devices. In *HCIR Workshop*, 2012.

[108] J. Huang and E. N. Efthimiadis. Analyzing and evaluating query reformulation strategies in web search logs. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*, pages 77–86, 2009.

[109] S. B. Huffman and M. Hochster. How well does result relevance predict session satisfaction? In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 567–574, 2007.

[110] R. Hull, P. Neaves, and J. Bedford-Roberts. Toward situated computing. *ISWC*, pages 146–153, 1997.

[111] S. Hyvönen, A. Gionis, and H. Mannila. Recurrent predictive models for sequence recurrent predictive models for sequence segmentation. In *IDA*, pages 195–206, 2007.

[112] S. Ieong, N. Mishra, E. Sadikov, and L. Zhang. Domain bias in web search. In *Proceedings of the ACM International conference on Web Search and Data Mining (WSDM)*, pages 55–64, 2012.

[113] Y. Inagaki, N. Sadagopan, G. Dupret, A. Dong, C. Liao, Y. Chang, and Z. Zheng. Session based click features for recency ranking. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2010.

[114] Y. Inagaki, N. Sadagopan, G. Dupret, C. L. A. Dong, Y. Chang, and Z. Zheng. Session based click features for recency ranking. In *Association for the Advancement of Artificial Intelligence*, 2010.

[115] P. Ingwersen and K. Järvelin. *The Turn: Integration of Information Seeking and Retrieval in Context (The Information Retrieval Series)*. Springer-Verlag New York, 2005.

[116] R. Jancey. Multidimensional group analysis. *Australian Journal of Botany*, 14(1):127–130, 1966.

[117] D. Jannach, Z. Karakaya, and F. Gedikli. Accuracy improvements for multi-

criteria recommender system. In *EC*, pages 674–689, 2012.

[118] B. J. Jansen, D. L. Booth, and A. Spink. Patterns of query reformulation during web searching. *JASIST*, 60(7):1358–1371, 2009.

[119] K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of ir techniques. *ACM Trans. Inf. Syst. (TOIS)*, 20(4):422–446, 2002.

[120] K. Järvelin, S. L. Price, L. M. L. Delcambre, and M. L. Nielsen. Discounted cumulated gain based evaluation of multiple-query ir sessions. In *Proceedings of the European Conference on Information Retrieval (ECIR)*, pages 4–15, 2008.

[121] J. Jiang, W. Jeng, and D. He. How do users respond to voice input errors?: lexical and phonetic query reformulation in voice search. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 143–152, 2013.

[122] J. Jiang, A. H. Awadallah, X. Shi, and R. W. White. Understanding and predicting graded search satisfaction. In *Proceedings of the ACM International conference on Web Search and Data Mining (WSDM)*, 2015.

[123] J. Jiang, A. Hassan Awadallah, R. Jones, U. Ozertem, I. Zitouni, R. G. Kulkarni, and O. Z. Khan. Automatic online evaluation of intelligent assistants. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 506–516, 2015.

[124] T. Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 133–142, 2002.

[125] T. Joachims, L. Granka, B. Pan, H. Hembrooke, and G. Gay. Accurately interpreting clickthrough data as implicit feedback. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 154–161, 2005.

[126] C. M. Jonker and A. Vollebregt. Iceberg: Exploiting context in information brokering agents. In *Proceedings of the International Workshop on Cooperative Information Agents*, pages 27–38, 2000.

[127] M. Kamvar, M. Kellar, R. Patel, and Y. Xu. Computers and iphones and mobile phones, oh my! In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 801–810, 2009.

[128] N. Kanhabua and K. Nørvåg. Learning to rank search results for time-sensitive queries. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*, pages 2463–2466, 2012.

[129] K. Kapoor, K. Subbian, J. Srivastava, and P. Schrater. Just in time recommendations: Modeling the dynamics of boredom in activity streams. In *Proceedings of the ACM International conference on Web Search and Data Mining (WSDM)*, pages 233–242, 2015.

[130] A. Karatzoglou, X. Amatriain, L. Baltrunas, and N. Oliver. Multiverse recommendation: n-dimensional tensor factorization for context-aware collaborative filtering. In *Proceedings of the ACM Conference on Recommender systems (RecSys)*, pages 79–86, 2010.

[131] D. Kelly. Methods for evaluating interactive information retrieval systems with users. *Foundations and Trends in Information Retrieval (FTIR)*, 3

(1-2):1–224, 2009.

[132] D. Kelly. When effort exceeds expectations: A theory of search task difficulty (keynote). In *SCST@ECIR*, volume 1338 of *CEUR Workshop Proceedings*, 2015.

[133] D. Kelly, J. Arguello, A. Edwards, and W. ching Wu. Development and evaluation of search tasks for iir experiments using a cognitive complexity framework. In *ICTIR*, pages 101–110, 2015.

[134] Y. Kim, A. Hassan, R. W. White, and Y.-M. Wang. Playing by the rules: mining query associations to predict search performance. In *Proceedings of the ACM International conference on Web Search and Data Mining (WSDM)*, pages 133–142, 2013.

[135] Y. Kim, A. Hassan, R. W. White, and I. Zitouni. Comparing client and server dwell time estimates for click-level satisfaction prediction. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 895–898, 2014.

[136] Y. Kim, A. Hassan, R. W. White, and I. Zitouni. Modeling dwell time to predict click-level satisfaction. In *Proceedings of the International ACM Conference on Web Search and Data Mining (WSDM)*, pages 193–202, 2014.

[137] J. Kiseleva. Grouping web users based on query log. In *Advances in Databases and Information Systems, Proceedings of the East European Conference (ADBIS)*, pages 184–190, 2008.

[138] J. Kiseleva. *Methods for web query analysis (in Russian)*. LAP LAMBERT Academic Publishing, 2011. ISBN 3847324551.

[139] J. Kiseleva. Context mining and integration into predictive web analytics. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 383–388, 2013.

[140] J. Kiseleva. Using contextual information to understand searching and browsing behavior. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval (Doctoral Consorcium)*, page 1059, 2015.

[141] J. Kiseleva, Q. Guo, E. Agichtein, D. Billsus, and W. Chai. Unsupervised query segmentation using click data: preliminary results. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 1131–1132, 2010.

[142] J. Kiseleva, E. Agichtein, and D. Billsus. Mining query structure from click data: a case study of product queries. In *Proceedings of the ACM Conference on Information and Knowledge Management (CIKM)*, pages 2217–2220, 2011.

[143] J. Kiseleva, H. T. Lam, M. Pechenizkiy, and T. Calders. Predicting current user intent with contextual markov models. In *Proceedings of the IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 391–398, 2013.

[144] J. Kiseleva, H. T. Lam, M. Pechenizkiy, and T. Calders. Discovering temporal hidden contexts in web sessions for user trail prediction. In *Companion Proceedings of the International Conference on World Wide Web (TempWeb)*, pages 1067–1074, 2013.

[145] J. Kiseleva, E. Crestan, R. Brigo, and R. Dittel. Modelling and detecting changes in user satisfaction. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*, pages 1449–1458, 2014.

[146] J. Kiseleva, A. Montes García, Y. Luo, J. Kamps, M. Pechenizkiy, and P. De Bra. Applying learning to rank techniques to contextual suggestions. In *Proceedings of the Text REtrieval Conference (TREC)*, 2014.

[147] J. Kiseleva, J. Kamps, and C. L. A. Clarke. Contextual search and exploration. *Communications in Computer and Information Science*, 2015.

[148] J. Kiseleva, J. Kamps, V. Nikulin, and N. Makarov. Behavioral dynamics from the SERP's perspective: What are failed SERPs and how to fix them? In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*, pages 1561–1570, 2015.

[149] J. Kiseleva, M. J. I. Müller, L. Bernardi, C. Davis, I. Kovacek, M. Stafseng Einarsen, J. Kamps, A. Tuzhilin, and D. Hiemstra. Where to go on your next trip? optimizing travel destinations based on user preferences. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 1097–1100, 2015.

[150] J. Kiseleva, A. Montes García, J. Kamps, and N. Spirin. The impact of technical domain expertise on search behavior and task outcome. In *Proceedings of WSDM Workshop on Query Understanding and Reformulation for Mobile and Web Search (QRUMS)*, 2016.

[151] J. Kiseleva, A. Tuzhilin, J. Kamps, M. J. I. Müller, L. Bernardi, C. Davis, I. Kovacek, M. Stafseng Einarsen, D. Hiemstra, and M. Pechenizkiy. Ranking travel destinations with contextual user profiles. Under Submission, 2016.

[152] J. Kiseleva, K. Williams, A. H. Awadallah, I. Zitouni, A. Crook, and T. Anastasakos. Predicting user satisfaction with intelligent assistants. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, 2016.

[153] J. Kiseleva, K. Williams, J. Jiang, A. H. Awadallah, I. Zitouni, A. Crook, and T. Anastasakos. Understanding user satisfaction with intelligent assistants. In *Proceedings of the ACM SIGIR Conference on Human Information Interaction and Retrieval (CHIIR)*, pages 121 – 130, 2016.

[154] D. Kluver and J. A. Konstan. Evaluating recommender behavior for new users. In *Proceedings of the ACM Conference on Recommender systems (RecSys)*, pages 121–128, 2014.

[155] R. Kohavi, A. Deng, R. Longbotham, and Y. Xu. Seven rules of thumb for web site experimenters. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 1857–1866, 2014.

[156] Y. Koren. Collaborative filtering with temporal dynamics. *Communications of the ACM*, 53(4):89–97, 2010.

[157] A. Kulkarni, J. Teevan, K. M. Svore, and S. T. Dumais. Understanding temporal query dynamics. In *Proceedings of the ACM International conference on Web Search and Data Mining (WSDM)*, pages 167–176, 2011.

[158] D. Lagun and E. Agichtein. Inferring searcher attention by jointly modeling user interactions and content salience. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 483–492, 2015.

[159] D. Lagun, C.-H. Hsieh, D. Webster, and V. Navalpakkam. Towards better measurement of attention and satisfaction in mobile search. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 113–122, 2014.

[160] K. Lakiotaki, N. F. Matsatsinis, and A. Tsoukias. Multicriteria user modeling in recommender systems. *IEEE Intelligent System*, 26(2):64–76, 2011.

[161] M. Lalmas, H. O'Brien, and E. Yom-Tov. Measuring user engagement. *Synthesis Lectures on Information Concepts, Retrieval, and Services*, 6(4): 1–132, 2014.

[162] H. T. Lam, J. Kiseleva, M. Pechenizkiy, and T. Calders. Decomposing a sequence into independent subsequences using compression algorithms. In *Proceeding of the ACM SIGKDD Workshop on Interactive Data Exploration and Analytic (IDEA)*, pages 67–75, 2014.

[163] J. Landis and G. Koch. The measurement of observer agreement for categorical data. *Biometrics*, pages 159–174, 1977.

[164] D. Lefortier, P. Serdyukov, and M. de Rijke. Online exploration for detecting shifts in fresh intent. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*, pages 589–598, 2014.

[165] J. Lehmann, M. Lalmas, G. Dupret, and R. A. Baeza-Yates. Online multitasking and user engagement. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*, pages 519–528, 2013.

[166] J. Li, S. B. Huffman, and A. Tokuda. Good abandonment in mobile and pc internet search. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 43–50, 2009.

[167] J. Li, P. Zhang, Y. Cao, P. Liu, and L. Guo. Efficient behavior targeting using svm ensemble indexing. In *ICDM*, pages 409 – 418, 2012.

[168] Z. Liao, Y. Song, L.-w. He, and Y. Huang. Evaluating the effectiveness of search task trails. In *Proceedings of the international conference on World Wide Web (WWW)*, pages 489–498, 2012.

[169] K. Liu and L. Tang. Large-scale behavioral targeting with a social twist. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*, pages 1815 – 1824, 2011.

[170] L. Liu, N. Mehandjiev, and D.-L. Xu. Multi-criteria service recommendation based on user criteria preferences. In *Proceedings of the ACM Conference on Recommender systems (RecSys)*, pages 77–84, 2011.

[171] Y. Liu, Y. Chen, J. Tang, J. Sun, M. Zhang, S. Ma, and X. Zhu. Different users, different opinions: Predicting search satisfaction with mouse movement information. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 493–502, 2015.

[172] J. Luo, X. Dong, and H. Yang. Session search by direct policy learning. In

*Proceedings of the International Conference on The Theory of Information Retrieval (ICTIR)*, pages 261–270, 2015. doi: 10.1145/2808194.2809461. URL `http://doi.acm.org/10.1145/2808194.2809461`.

[173] J. Luo, S. Zhang, X. Dong, and H. Yang. Designing states, actions, and rewards for using pomdp in session search. In *Proceedings of the European Conference on Information Retrieval (ECIR)*, pages 526–537, 2015.

[174] H. Ma, H. Cao, Q. Yang, E. Chen, and J. Tian. A habit mining approach for discovering similar mobile users. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 231–240, 2012.

[175] O. Mazhelis, I. Zliobaite, and M. Pechenizkiy. Context-aware personal route recognition. In *Proceedings of The International Conference on Discovery Science (DS)*, pages 211–235, 2011.

[176] M. F. McTear. Spoken dialogue technology: enabling the conversational user interface. *ACM Computing Surveys (CSUR)*, 34(1):90–169, 2002.

[177] G. D. Montanez, R. W. White, and X. Huang. Cross-device search. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*, pages 1669–1678, 2014.

[178] M. Negri, M. Turchi, J. G. C. de Souza, and D. Falavigna. Quality estimation for automatic speech recognition. In *COLING*, pages 1813–1823, 2014.

[179] A. Noulas and M. Stafseng Einarsen. User engagement through topic modelling in travel. In *Second Workshop on User Engagement Optimization*, 2014.

[180] C. Palmisano, A. Tuzhilin, and M. Gorgoglione. Using context to improve predictive modeling of customers in personalization applications. *TKDE*, 20(11):1535–1549, November 2008.

[181] U. Panniello, A. Tuzhilin, M. Gorgoglione, C. Palmisano, and A. Pedone. Experimental comparison of pre- vs. post-filtering approaches in context-aware recommender systems. In *Proceedings of the ACM Conference on Recommender systems (RecSys)*, pages 265–268, 2009.

[182] L. Philips. Hanging on the metaphone. *Computer Language*, 7(12):39–44, 1990.

[183] C. Prahalad. Beyond crm: Predicts customer context is the next big thing. *AMA MwWorld*, 2004.

[184] K. Radinsky, S. Davidovich, and S. Markovitch. Predicting the news of tomorrow using patterns in web search queries. In *WI*, 2008.

[185] K. Radinsky, K. Svore, S. T. Dumais, J. Teevan, A. Bocharov, and E. Horvitz. Modeling and predicting behavioral dynamics on the web. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 599–608, 2012.

[186] K. Radinsky, K. M. Svore, S. T. Dumais, M. Shokouhi, J. Teevan, A. Bocharov, and E. Horvitz. Behavioral dynamics on the web: Learning, modeling, and prediction. *ACM Transactions on Information Systems (TOIS)*, 31(3):16, 2013.

[187] A. M. Rashid, I. Albert, D. Cosley, S. K. Lam, S. M. McNee, J. A. Konstan, and J. Riedl. Getting to know you: Learning new user preferences in

recommender systems. In *IUI*, pages 127–134, 2002.
[188] S. Rendle. Factorization machines. In *ICDM*, pages 995–1000, 2009.
[189] S. Rendle, Z. Gantner, C. Freudenthaler, and L. Schmidt-Thieme. Fast context-aware recommendations with factorization machines. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 635–644, 2011.
[190] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. Grouplens: An open architecture for collaborative filtering of netnews. In *CSCW*, pages 175–186, 1994.
[191] K. Rodden, X. Fu, A. Aula, and I. Spiro. Eye-mouse coordination patterns on web search results pages. In *Proceeding of the CHI Extended Abstracts*, pages 2997–3002, 2008.
[192] D. Ron, Y. Singer, and N. Tishby. The power of amnesia: Learning probabilistic automata with variable memory length. *Machine Learning (ML)*, 25(2-3):117–149, 1996.
[193] D. E. Rose and D. Levinson. Understanding user goals in web search. In *Proceedings of the International Conference on World Wide Web, (WWW)*, pages 13–19, 2004. doi: 10.1145/988672.988675. URL `http://doi.acm.org/10.1145/988672.988675`.
[194] P. J. Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *JCAM*, 20:53–65, 1987.
[195] N. Samaan and A. Karmouch. A mobility prediction architecture based on contextual knowledge and spatial conceptual maps. *IEEE Trans. Mob. Comput. (TMC)*, 4(6):537–551, 2005.
[196] T. Saracevic. Relevance: A review of and a framework for the thinking on the notion in information science. *Journal of the American Society for Information Science*, 26:321–343, 1975.
[197] T. Saracevic. Relevance: A review of the literature and a framework for thinking on the notion in information science. part ii: nature and manifestations of relevance. *JASIST (JASIS)*, 58(13):1915–1933, 2007.
[198] T. Saracevic. Relevance: A review a the literature and a framework for thinking on the notion in information science. part iii: Behavior and effects of relevance. *JASIST (JASIS)*, 58(13):2126–2144, 2007.
[199] T. Saracevic, P. B. Kantor, A. Y. Chamis, and D. Trivison. A study of information seeking and retrieving. I. background and methodology. II. users, questions and effectiveness. III. searchers, searches, overlap. *Journal of the American Society for Information Science and Technology*, 39:161–176; 177–196; 197–216, 1988.
[200] M. Saveski and A. Mantrach. Item cold-start recommendations: Learning local collective embeddings. In *Proceedings of the ACM Conference on Recommender systems (RecSys)*, pages 89–96, 2014.
[201] A. I. Schein, A. Popescul, L. H. Ungar, and D. M. Pennock. Methods and metrics for cold-start recommendations. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 253–260, 2002.
[202] J. C. Schlimmer and R. H. Granger. Beyond incremental processing: Track-

ing concept drift. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 1986.

[203] A. Schmidt, M. Beigl, and H.-W. Gellersen. There is more to context than location. *Computers & Graphics*, 23(6):893–901, 1999.

[204] S. Sedhain, S. Sanner, D. Braziunas, L. Xie, and J. Christensen. Social collaborative filtering for cold-start recommendations. In *Proceedings of the ACM Conference on Recommender systems (RecSys)*, pages 345–348, 2014.

[205] U. Shardanand and P. Maes. Social information filtering: Algorithms for automating 'word of mouth'. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI)*, pages 210–217, 1995.

[206] X. Shen, S. T. Dumais, and E. Horvitz. Analysis of topic dynamics in web search. In *Proceedings of the International Conference on World Wide Web (WWW)*, volume 1112–1113, 2005 (Special interest tracks and posters).

[207] Y. Shi, A. Karatzoglou, L. Baltrunas, M. Larson, A. Hanjalic, and N. Oliver. Tfmap: optimizing map for top-n context-aware recommendation. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 155–164, 2012.

[208] Y. Shi, A. Karatzoglou, L. Baltrunas, M. Larson, and A. Hanjalic. Cars2: Learning context-aware representations for context-aware recommendations. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*, pages 291–300, 2014.

[209] M. Shokouhi. Detecting seasonal queries by time-series analysis. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 1171–1172, 2011.

[210] M. Shokouhi. Learning to personalize query auto-completion. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 103–112, 2013. doi: 10.1145/2484028.2484076. URL `http://doi.ACM.org/10.1145/2484028.2484076`.

[211] M. Shokouhi and Q. Guo. From queries to cards: Re-ranking proactive card recommendations based on reactive search history. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 695–704, 2015.

[212] M. Shokouhi and K. Radinsky. Time-sensitive query auto-completion. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 601–610, 2012.

[213] A. Shtok, O. Kurland, D. Carmel, F. Raiber, and G. Markovits. Predicting query performance by query-drift estimation. *ACM Transactions on Information Systems (TOIS)*, 30:11:1–11:35, 2012.

[214] J. Simoes, J. Kiseleva, E. Sivogolovko, and B. Novikov. Exploring influence and interests among users within social networks. In *Computational Social Networks*, pages 177–206. Springer London, 2012.

[215] M. Song, C. W. Günther, and W. M. P. van der Aalst. Trace clustering in process mining. In *Procceedings of Business Process Management Workshops, BPM International Workshops*, pages 109–120, 2008. doi: 10.1007/978-3-642-00328-8_11. URL `http://dx.doi.org/10.1007/978-3-642-00328-8_11`.

[216] Y. Song, X. Shi, and X. Fu. Evaluating and predicting user engagement change with degraded search relevance. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 1213–1224, 2013.

[217] Y. Song, X. Shi, R. White, and A. H. Awadallah. Context-aware web search abandonment prediction. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 93–102, 2014.

[218] N. Spirin, M. Kuznetsov, J. Kiseleva, Y. Spirin, and P. Izhutov. Relevance-aware filtering of tuples sorted by an attribute value via direct optimization of metrics. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 979–982, 2015.

[219] S. Stamou and E. N. Efthimiadis. Interpreting user inactivity on search results. In *Proceedings of the European Conference on Information Retrieval (ECIR)*, volume 5993, pages 100–113, 2010.

[220] Statcounter Global Stats. Statcounter global stats, 2015. `http://gs.statcounter.com/#mobile_vs_desktop-ww-monthly-201109-201509`.

[221] D. Stern, R. Herbrich, and T. Graepel. Matchbox: Large scale online bayesian recommendations. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 111–120, 2009.

[222] J. Z. Sun, K. R. Varshney, and K. Subbian. Dynamic matrix factorization: A state space approach. In *ICASSP*, pages 1897–1900, 2012.

[223] D. Tang, A. Agarwal, D. O'Brien, and M. Meyer. Overlapping experiment infrastructure: More, better, faster experimentation. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 17–26, Washington, DC, 2010.

[224] J. Tang, H. Gao, X. Hu, and H. Liu. Context-aware review helpfulness rating prediction. In *Proceedings of the ACM Conference on Recommender systems (RecSys)*, pages 1–8, 2013.

[225] L. Tang, Y. Jiang, L. Li, and T. Li. Ensemble contextual bandits for personalized recommendation. In *Proceedings of the ACM Conference on Recommender systems (RecSys)*, pages 73–80, 2014.

[226] M. Tavakol and U. Brefeld. Factored mdps for detecting topics of user sessions. In *Proceedings of the ACM Conference on Recommender systems (RecSys)*, pages 33–40, 2014. doi: 10.1145/2645710.2645739.

[227] TREC. Text retrieval conference: Contextual suggestion track. https://sites.google.com/site/treccontext/, 2015.

[228] G. Tür. Extending boosting for large scale spoken language understanding. *Machine Learning (ML)*, 69(1):55–74, 2007.

[229] G. Tür, Y.-Y. Wang, and D. Z. Hakkani-Tür. Techware: Spoken language understanding resources [best of the web]. *IEEE Signal Process. Mag. (SPM)*, 30(3):187–189, 2013.

[230] G. Tür, Y.-Y. Wang, and D. Z. Hakkani-Tür. Understanding spoken language. *Computing Handbook*, 3rd ed(41):1–17, 2014.

[231] P. Turney. Exploiting context when learning to classify. In *ECML*, pages 402–407, 1993.

[232] P. D. Turney. The management of context-sensitive features: A review of

strategies. *CoRR*, cs.LG/0212037, 2002. URL `http://arxiv.org/abs/cs.LG/0212037`.

[233] P. D. Turney. The identification of context-sensitive features: A formal definition of context for concept learning. *CoRR*, cs.LG/0212038, 2002. URL `http://arxiv.org/abs/cs.LG/0212038`.

[234] P. Vakkari. Task-based information searching. *ARIST*, 37:413–464, 2003.

[235] B. Vargas-Govea, J. G. González-Serna, and R. Ponce-Medellín. Effects of relevant contextual features in the performance of a restaurant recommender system. In *Proceedings of the Workshop on Context-Aware Recommender Systems (CARS)*, 2011.

[236] W. Wahlster. Smartkom: Foundations of multimodal dialogue systems. *Springer*, 2006.

[237] Y. Wang and E. Agichtein. Query ambiguity revisited: Clickthrough measures for distinguishing informational and ambiguous queries. In *HLT-NAACL*, pages 361–364, 2010.

[238] R. Want, A. Hopper, V. Falcão, and J. Gibbons. The active badge location system. *ACM Transactions on Information Systems (TOIS)*, 10(1):91–202, 1992.

[239] R. W. White and S. T. Dumais. Characterizing and predicting search engine switching behavior. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*, pages 87–96, 2009.

[240] R. W. White, M. Richardson, and W.-t. Yih. Questions vs. queries in informational search tasks. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 135–136, 2015.

[241] G. Widmer and M. Kubat. Learning in the presence of concept drift and hidden contexts. *Machine Learning (ML)*, 23(1):69–101, 1996.

[242] B. Wildemuth, L. Freund, and E. G. Toms. Untangling search task complexity and difficulty in the context of interactive information retrieval studies. *Journal of Documentation*, 70:1118–1140, 2014.

[243] F. M. J. Willems. The context-tree weighting method: Extensions. *IEEE Transactions on Information Theory (TIT)*, 44(2):792–798, 1998.

[244] K. Williams, J. Kiseleva, A. Crook, I. Zitouni, A. H. Awadallah, and M. Khabsa. Is this your final answer? evaluating the effect of answers on good abandonment in mobile search. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, 2016.

[245] K. Williams, J. Kiseleva, A. C. Crook, I. Zitouni, A. H. Awadallah, and M. Khabsa. Detecting good abandonment in mobile search. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 495 – 505, 2016.

[246] T. Wilson. Models in information behaviour research. *Journal of Documentation*, 55(3):249 – 270, 1999.

[247] B. Xiang, D. Jiang, J. Pei, X. Sun, E. Chen, and H. Li. Context-aware ranking in web search. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, 2010.

[248] H. Yang, D. Guan, and S. Zhang. The query change model: Modeling session

search as a markov decision process. *ACM Transactions on Information Systems (TOIS)*, 33(4), 2015.

[249] E. Yilmaz, M. Verma, N. Craswell, F. Radlinski, and P. Bailey. Relevance and effort: An analysis of document utility. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*, pages 91–100, 2014.

[250] Y. Yue, R. Patel, and H. Roehrig. Beyond position bias: Examining result attractiveness as a source of presentation bias in clickthrough data. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 1011–1018, 2010.

[251] H. Zhu, H. Cao, E. Chen, H. Xiong, and J. Tian. Exploiting enriched contextual information for mobile app classification. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*, pages 1617–1621, 2012.

[252] I. Zliobaite. Learning under concept drift: an overview. *CoRR*, abs/1010.4784, 2010. URL `http://arxiv.org/abs/1010.4784`.

[253] I. Zliobaite. Identifying hidden contexts in classification. In *Proceedings of the Pacific-Asia Conference Advances in Knowledge Discovery and Data Mining (PAKDD) Part I*, pages 277–288, 2011. doi: 10.1007/978-3-642-20841-6_23. URL `http://dx.doi.org/10.1007/978-3-642-20841-6_23`.

[254] I. Zliobaite, J. Bakker, and M. Pechenizkiy:. Towards context aware food sales prediction. In *ICDM Workshops*, pages 94–99, 2009.

[255] I. Zliobaite, J. Bakker, and M. Pechenizkiy. Beating the baseline prediction in food sales: How intelligent an intelligent predictor is? *Expert Syst. Appl. (ESWA)*, 39(1):806–815, 2012.

[256] I. Zukerman and D. W. Albrecht. Predictive statistical models for user modeling. *User Modeling and User-Adapted Interaction*, 11:5–18, 2001.

# Index

# Summary

**Using Contextual Information to Understand Searching and Browsing Behavior**

Modern search still relies on the query-response paradigm, which is characterized by a sharp contrast between the richness of data in the index, and the relative poverty of information in the query, usually expressed in a few keywords to capture a complex need. This is particularly true in online search services, where the same query may be observed from many users, with considerable variations in their search intents. Contextual information is the obvious route to try to restore the balance, and behavioral data related to user's searching and browsing activities provides new opportunities to model contextual aspects of user needs. The importance of contextual information in search applications has been recognized by researchers and practitioners in many disciplines, including recommendation systems, information retrieval, ubiquitous and mobile computing, and marketing. Context-aware systems adapt to users' operations and thus aim at improving the usability and effectiveness by taking context into account. In this thesis we consider two types of behavior: *searching*, when users are issuing queries and we are trying to improve the search engine results page (SERP) by taking the context of sessions into account, and *browsing*, when users are surfing a website and we are predicting their movements using context. Finding ways to better leverage contextual information and make search context-aware holds the promise to dramatically improve the search experience of users. We conducted a series of studies to discover, model and use contextual information in order to understand and improve users' searching and browsing behavior on the web.

Our main contributions are the following: First, we focused on the system's centric view of context-aware information interaction, and defined a general framework for discovering context, and the notions of optimal contextual models and useful contextual models. Second, we studied the impact of behavioral aspects of users, as captured in search trails, discovering groups of similar trails (and hence similar users) based on next action prediction using data from an online study choice portal, and discovering clusters of similar contextual endorsements using data from booking.com. Third, we studied methods to infer user satisfaction for voice-controlled intelligent assistants. We considered two important scenarios of intelligent assistants use: web search and search dialogs. Web search on mobile devices leads frequently to good abandonment where users infer the required information without clicking on search results. Search dialogues are a new way of interaction with intelligent assistants where users use voice and a system is able to keep a context of the conversation. We showed that touch interaction is useful to determine user satisfaction. Fourth, we looked at the impact of changes in behavioral aspects over time, using changes in frequency of query revisions and SAT/DSAT clicks to detect changes in user satisfaction and drifts in query intent,

and identify revision terms and URLs to be used in query completion or SERP re-ranking, using data from Microsoft Bing and Yandex. Our results capture important aspects of context under the realistic conditions of different online search services, aiming to ensure that our scientific insights and solutions transfer to the operational settings of real world applications.

# Samenvatting


**Gebruik van Contextuele Informatie om Zoek- en Browse-gedrag te Begrijpen**

Moderne zoekmachines zijn nog steeds afhankelijk van het query-response paradigma, dat wordt gekenmerkt door een scherp contrast tussen de rijkdom van de gegevens in de index en de relatieve armoede van informatie in de query, die meestal in slechts een paar trefwoorden een complex behoefte uitdrukt. Dit geldt met name voor online zoekdiensten, waar een zelfde zoekvraag kan worden gesteld door een groot aantal gebruikers met onderling verschillende informatiebehoeften. Contextuele informatie is de voor de hand liggende manier om te proberen om het evenwicht te herstellen, en interactielogs met zoek- en browse-gedrag van gebruikers bieden nieuwe mogelijkheden om contextuele aspecten van de informatiebehoeften van de gebruiker te modelleren. Het belang van de contextuele informatie is onderkend door onderzoekers uit vele disciplines, waaronder aanbevelingssystemen, informatie retrieval, mobiel zoeken, en marketing. Contextgevoelige systemen passen zich aan aan activiteiten van gebruikers en zijn dus gericht op een verbetering van de effectiviteit en bruikbaarheid door rekening te houden met context. In dit proefschrift beschouwen we twee soorten gedrag: zoeken, wanneer gebruikers zoekvragen stellen en we proberen om de zoekmachine resultaten pagina (SERP) te verbeteren door de context van sessies mee te nemen, en browsing, wanneer gebruikers surfen op een website en we proberen hun klikpaden te voorspellen met behulp van context. Manieren die contextuele informatie beter gebruiken hebben de potentie om de zoekervaring van gebruikers drastisch te verbeteren. We hebben een reeks van studies uitgevoerd om contextuele informatie te ontdekken, modeleren en te gebruiken voor de verbetering van, en een beter inzicht in, gebruikers zoek- en surfgedrag op het web.

Onze belangrijkste bijdragen zijn de volgende: Ten eerste hebben we ons gericht op systeemkant van contextgevoelige informatie-interactie, de definitie van een algemeen raamwerk voor het ontdekken van context, en van de begrippen van optimale en nuttige contextuele modellen. Ten tweede hebben we ons gericht op de gebruikerskant door een analyse van gebruikersgedrag zoals vastgelegd in zoekpaden, voor het ontdekken van groepen van vergelijkbare zoekpaden (en dus vergelijkbare gebruikers) gebaseerd op een volgende-actie-voorspelling voor de interactielogs van een online studiekeuze website, en het ontdekken van clusters van soortgelijke contextuele aanbevelingen met de interactielogs van booking.com. Ten derde bestudeerden we methoden om de tevredenheid van een gebruiker van spraakgestuurde intelligente assistenten af te leiden. We onderzoeken twee belangrijke scenarios van gebruik: web zoeken en zoekdialogen. Web zoeken op mobiele apparaten leidt vaak tot "good abandonment", waarbij de gebruikers de informatie vinden zonder door te hoeven klikken op de zoekresultaten. Een zoekdialoog is een nieuwe manier van interactie met een intelligente assistent, die gebruik maakt

van spraak en in staat is om de context van het gesprek bij te houden. We tonen aan dat aanrakingsinteractie nuttig is om de tevredenheid van de gebruikers te bepalen. Ten vierde hebben we gekeken naar de impact van veranderingen in gedragsaspecten over verloop van tijd, waarbij veranderingen in de frequentie van query-revisies en SAT/DSAT-klikken helpen om veranderingen in gebruikerstevredenheid en query-intentie te detecteren, en query-revisie-termen en -URL's helpen voor query-suggestie of het verbeteren van de SERP, met behulp van gegevens van Microsoft Bing en Yandex. Onze resultaten behandelen belangrijke aspecten van context onder realistische omstandigheden van verschillende online zoekdiensten, met als doel om ervoor te zorgen dat onze wetenschappelijke inzichten en oplossingen toepasbaar zijn onder de realistische omstandigheden van online toepassingen in de echte wereld.

# Curriculum Vitae

## Short Resume

Julia Kiseleva was born on 9 December 1984 in Kaliningrad, Russia. After finishing the lyceum "Ganzeiskaya Ladya" in 2002 in Kaliningrad, Russia, she studied Applied Mathematics at Saint-Petersburg State University in Saint-Petersburg, Russia. In 2007 she graduated within the "Game Theory and Statistics" research group. Afterwards she has been working as Research Engineer in various of companies including Ebay.com, Hewlett Packard Labs, Yandex.ru and others.

In March 2012 she started on a PhD project at Eindhoven University of Technology, Eindhoven, the Netherlands. The results of this project are presented in this dissertation. While doing her PhD studies, Julia has done three inductrial internships, at Microsoft Bing, at Booking.com, and at Microsoft Research, that have resulted in number of publications. Julia served as program committee member at SIGIR 2015, CIKM 2015, ACL 2016, SIGIR 2016, RUSSIR 2015, and others. Additionally, Julia co-organized the Contextual Suggestion Track at Text REtrieval Conference (TREC) in 2015 and 2016.

In February 2016, Julia has been awarded an STW grant for a project called "Understanding and Predicting User Satisfaction".

## List of Publications

Selected publications:

**[29]** L. Bernardi, J. Kamps, J. Kiseleva, and M. J. I. Müller. The continuous cold start problem in e-commerce recommender systems. In *Proceedings of the Workshop on New Trends on Content-Based Recommender Systems co-located with ACM Conference on Recommender Systems*, pages 30–33, 2015.

**[60]** A. Dean-Hall, C. L. A. Clarke, J. Kamps, and J. Kiseleva. Online evaluation of point-of-interest recommendation systems. In *Proceedings of SCST@ECIR*, 2015.

**[61]** A. Dean-Hall, C. L. A. Clarke, J. Kamps, J. Kiseleva, and E. M. Voorhees. Overview of the TREC 2015 contextual suggestion track. In *Proceedings of the Text REtrieval Conference (TREC)*, 2015.

**[93]** H. Hashemi, C. L. A. Clarke, A. Dean-Hall, J. Kamps, and J. Kiseleva. On the reusability of open test collections. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 827–830, 2015.

**[94]** S. H. Hashemi, C. L. A. Clarke, A. Dean-Hall, J. Kamps, and J. Kiseleva. An easter egg hunting approach to test collection building in dynamic domains. 2016. EVIA@NTCIR.

**[146]** J. Kiseleva, A. Montes García, Y. Luo, J. Kamps, M. Pechenizkiy, and P. De Bra. Applying learning to rank techniques to contextual suggestions. In *Proceedings of the Text REtrieval Conference (TREC)*, 2014.

**[147]** J. Kiseleva, J. Kamps, and C. L. A. Clarke. Contextual search and exploration. *Communications in Computer and Information Science*, 2015.

**[150]** J. Kiseleva, A. Montes García, J. Kamps, and N. Spirin. The impact of technical domain expertise on search behavior and task outcome. In *Proceedings of WSDM Workshop on Query Understanding and Reformulation for Mobile and Web Search (QRUMS)*, 2016.

**[148]** J. Kiseleva, J. Kamps, V. Nikulin, and N. Makarov. Behavioral dynamics from the SERP's perspective: What are failed SERPs and how to fix them? In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*, pages 1561–1570, 2015.

**[145]** J. Kiseleva, E. Crestan, R. Brigo, and R. Dittel. Modelling and detecting changes in user satisfaction. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*, pages 1449–1458, 2014.

**[153]** J. Kiseleva, K. Williams, J. Jiang, A. H. Awadallah, I. Zitouni, A. Crook, and T. Anastasakos. Understanding user satisfaction with intelligent assistants. In *Proceedings of the ACM SIGIR Conference on Human Information Interaction and Retrieval (CHIIR)*, pages 121 – 130, 2016.

**[152]** J. Kiseleva, K. Williams, A. H. Awadallah, I. Zitouni, A. Crook, and T. Anastasakos. Predicting user satisfaction with intelligent assistants. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, 2016.

**[149]** J. Kiseleva, M. J. I. Müller, L. Bernardi, C. Davis, I. Kovacek, M. Stafseng Einarsen, J. Kamps, A. Tuzhilin, and D. Hiemstra. Where to go on your next trip? optimizing travel destinations based on user preferences. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 1097–1100, 2015.

**[143]** J. Kiseleva, H. T. Lam, M. Pechenizkiy, and T. Calders. Predicting current user intent with contextual markov models. In *Proceedings of the IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 391–398, 2013.

**[144]** J. Kiseleva, H. T. Lam, M. Pechenizkiy, and T. Calders. Discovering temporal hidden contexts in web sessions for user trail prediction. In *Companion*

*Proceedings of the International Conference on World Wide Web (TempWeb)*, pages 1067–1074, 2013.

**[139]** J. Kiseleva. Context mining and integration into predictive web analytics. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 383–388, 2013.

**[140]** J. Kiseleva. Using contextual information to understand searching and browsing behavior. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval (Doctoral Consorcium)*, page 1059, 2015.

**[137]** J. Kiseleva. Grouping web users based on query log. In *Advances in Databases and Information Systems, Proceedings of the East European Conference (ADBIS)*, pages 184–190, 2008.

**[141]** J. Kiseleva, Q. Guo, E. Agichtein, D. Billsus, and W. Chai. Unsupervised query segmentation using click data: preliminary results. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 1131–1132, 2010.

**[142]** J. Kiseleva, E. Agichtein, and D. Billsus. Mining query structure from click data: a case study of product queries. In *Proceedings of the ACM Conference on Information and Knowledge Management (CIKM)*, pages 2217–2220, 2011.

**[138]** J. Kiseleva. *Methods for web query analysis (in Russian)*. LAP LAMBERT Academic Publishing, 2011. ISBN 3847324551.

**[162]** H. T. Lam, J. Kiseleva, M. Pechenizkiy, and T. Calders. Decomposing a sequence into independent subsequences using compression algorithms. In *Proceeding of the ACM SIGKDD Workshop on Interactive Data Exploration and Analytic (IDEA)*, pages 67–75, 2014.

**[214]** J. Simoes, J. Kiseleva, E. Sivogolovko, and B. Novikov. Exploring influence and interests among users within social networks. In *Computational Social Networks*, pages 177–206. Springer London, 2012.

**[218]** N. Spirin, M. Kuznetsov, J. Kiseleva, Y. Spirin, and P. Izhutov. Relevance-aware filtering of tuples sorted by an attribute value via direct optimization of metrics. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 979–982, 2015.

**[245]** K. Williams, J. Kiseleva, A. C. Crook, I. Zitouni, A. H. Awadallah, and M. Khabsa. Detecting good abandonment in mobile search. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 495 – 505, 2016.

**[244]** K. Williams, J. Kiseleva, A. Crook, I. Zitouni, A. H. Awadallah, and M. Khabsa. Is this your final answer? evaluating the effect of answers on good abandonment in mobile search. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, 2016.

# SIKS Dissertation Series

**1998**

1 Johan van den Akker (CWI) *DEGAS: An Active, Temporal Database of Autonomous Objects*
2 Floris Wiesman (UM) *Information Retrieval by Graphically Browsing Meta-Information*
3 Ans Steuten (TUD) *A Contribution to the Linguistic Analysis of Business Conversations*
4 Dennis Breuker (UM) *Memory versus Search in Games*
5 E. W. Oskamp (RUL) *Computerondersteuning bij Straftoemeting*

**1999**

1 Mark Sloof (VUA) *Physiology of Quality Change Modelling: Automated modelling of*
2 Rob Potharst (EUR) *Classification using decision trees and neural nets*
3 Don Beal (UM) *The Nature of Minimax Search*
4 Jacques Penders (UM) *The practical Art of Moving Physical Objects*
5 Aldo de Moor (KUB) *Empowering Communities: A Method for the Legitimate User-Driven*
6 Niek J. E. Wijngaards (VUA) *Re-design of compositional systems*
7 David Spelt (UT) *Verification support for object database design*
8 Jacques H. J. Lenting (UM) *Informed Gambling: Conception and Analysis of a Multi-Agent Mechanism*

**2000**

1 Frank Niessink (VUA) *Perspectives on Improving Software Maintenance*
2 Koen Holtman (TUe) *Prototyping of CMS Storage Management*
3 Carolien M. T. Metselaar (UvA) *Sociaal-organisatorische gevolgen van kennistechnologie*
4 Geert de Haan (VUA) *ETAG, A Formal Model of Competence Knowledge for User Interface*
5 Ruud van der Pol (UM) *Knowledge-based Query Formulation in Information Retrieval*
6 Rogier van Eijk (UU) *Programming Languages for Agent Communication*
7 Niels Peek (UU) *Decision-theoretic Planning of Clinical Patient Management*
8 Veerle Coupé (EUR) *Sensitivity Analyis of Decision-Theoretic Networks*
9 Florian Waas (CWI) *Principles of Probabilistic Query Optimization*
10 Niels Nes (CWI) *Image Database Management System Design Considerations, Algorithms and Architecture*
11 Jonas Karlsson (CWI) *Scalable Distributed Data Structures for Database Management*

**2001**

1 Silja Renooij (UU) *Qualitative Approaches to Quantifying Probabilistic Networks*
2 Koen Hindriks (UU) *Agent Programming Languages: Programming with Mental Models*
3 Maarten van Someren (UvA) *Learning as problem solving*
4 Evgueni Smirnov (UM) *Conjunctive and Disjunctive Version Spaces with Instance-Based Boundary Sets*
5 Jacco van Ossenbruggen (VUA) *Processing Structured Hypermedia: A Matter of Style*
6 Martijn van Welie (VUA) *Task-based User Interface Design*
7 Bastiaan Schonhage (VUA) *Diva: Architectural Perspectives on Information Visualization*
8 Pascal van Eck (VUA) *A Compositional Semantic Structure for Multi-Agent Systems Dynamics*
9 Pieter Jan 't Hoen (RUL) *Towards Distributed Development of Large Object-Oriented Models*
10 Maarten Sierhuis (UvA) *Modeling and Simulating Work Practice*
11 Tom M. van Engers (VUA) *Knowledge Management*

**2002**

1 Nico Lassing (VUA) *Architecture-Level Modifiability Analysis*

2 Roelof van Zwol (UT) *Modelling and searching web-based document collections*
3 Henk Ernst Blok (UT) *Database Optimization Aspects for Information Retrieval*
4 Juan Roberto Castelo Valdueza (UU) *The Discrete Acyclic Digraph Markov Model in Data Mining*
5 Radu Serban (VUA) *The Private Cyberspace Modeling Electronic*
6 Laurens Mommers (UL) *Applied legal epistemology: Building a knowledge-based ontology of*
7 Peter Boncz (CWI) *Monet: A Next-Generation DBMS Kernel For Query-Intensive*
8 Jaap Gordijn (VUA) *Value Based Requirements Engineering: Exploring Innovative*
9 Willem-Jan van den Heuvel (KUB) *Integrating Modern Business Applications with Objectified Legacy*
10 Brian Sheppard (UM) *Towards Perfect Play of Scrabble*
11 Wouter C. A. Wijngaards (VUA) *Agent Based Modelling of Dynamics: Biological and Organisational Applications*
12 Albrecht Schmidt (UvA) *Processing XML in Database Systems*
13 Hongjing Wu (TUe) *A Reference Architecture for Adaptive Hypermedia Applications*
14 Wieke de Vries (UU) *Agent Interaction: Abstract Approaches to Modelling, Programming and Verifying Multi-Agent Systems*
15 Rik Eshuis (UT) *Semantics and Verification of UML Activity Diagrams for Workflow Modelling*
16 Pieter van Langen (VUA) *The Anatomy of Design: Foundations, Models and Applications*
17 Stefan Manegold (UvA) *Understanding, Modeling, and Improving Main-Memory Database Performance*

**2003**

1 Heiner Stuckenschmidt (VUA) *Ontology-Based Information Sharing in Weakly Structured Environments*
2 Jan Broersen (VUA) *Modal Action Logics for Reasoning About Reactive Systems*
3 Martijn Schuemie (TUD) *Human-Computer Interaction and Presence in Virtual Reality Exposure Therapy*
4 Milan Petkovic (UT) *Content-Based Video Retrieval Supported by Database Technology*
5 Jos Lehmann (UvA) *Causation in Artificial Intelligence and Law: A modelling approach*
6 Boris van Schooten (UT) *Development and specification of virtual environments*
7 Machiel Jansen (UvA) *Formal Explorations of Knowledge Intensive Tasks*
8 Yongping Ran (UM) *Repair Based Scheduling*
9 Rens Kortmann (UM) *The resolution of visually guided behaviour*
10 Andreas Lincke (UvT) *Electronic Business Negotiation: Some experimental studies on the interaction between medium, innovation context and culture*
11 Simon Keizer (UT) *Reasoning under Uncertainty in Natural Language Dialogue using Bayesian Networks*
12 Roeland Ordelman (UT) *Dutch speech recognition in multimedia information retrieval*
13 Jeroen Donkers (UM) *Nosce Hostem: Searching with Opponent Models*
14 Stijn Hoppenbrouwers (KUN) *Freezing Language: Conceptualisation Processes across ICT-Supported Organisations*
15 Mathijs de Weerdt (TUD) *Plan Merging in Multi-Agent Systems*
16 Menzo Windhouwer (CWI) *Feature Grammar Systems: Incremental Maintenance of Indexes to Digital Media Warehouses*
17 David Jansen (UT) *Extensions of Statecharts with Probability, Time, and Stochastic Timing*
18 Levente Kocsis (UM) *Learning Search Decisions*

**2004**

1 Virginia Dignum (UU) *A Model for Organizational Interaction: Based on Agents, Founded in Logic*
2 Lai Xu (UvT) *Monitoring Multi-party Contracts for E-business*
3 Perry Groot (VUA) *A Theoretical and Empirical Analysis of Approximation in Symbolic Problem Solving*
4 Chris van Aart (UvA) *Organizational Principles for Multi-Agent Architectures*
5 Viara Popova (EUR) *Knowledge discovery and monotonicity*
6 Bart-Jan Hommes (TUD) *The Evaluation of Business Process Modeling Techniques*
7 Elise Boltjes (UM) *Voorbeeldig onderwijs: voorbeeldgestuurd onderwijs, een opstap naar abstract denken, vooral voor meisjes*

8 Joop Verbeek (UM) *Politie en de Nieuwe Internationale Informatiemarkt, Grensregionale politiële gegevensuitwisseling en digitale expertise*
9 Martin Caminada (VUA) *For the Sake of the Argument: explorations into argument-based reasoning*
10 Suzanne Kabel (UvA) *Knowledge-rich indexing of learning-objects*
11 Michel Klein (VUA) *Change Management for Distributed Ontologies*
12 The Duy Bui (UT) *Creating emotions and facial expressions for embodied agents*
13 Wojciech Jamroga (UT) *Using Multiple Models of Reality: On Agents who Know how to Play*
14 Paul Harrenstein (UU) *Logic in Conflict. Logical Explorations in Strategic Equilibrium*
15 Arno Knobbe (UU) *Multi-Relational Data Mining*
16 Federico Divina (VUA) *Hybrid Genetic Relational Search for Inductive Learning*
17 Mark Winands (UM) *Informed Search in Complex Games*
18 Vania Bessa Machado (UvA) *Supporting the Construction of Qualitative Knowledge Models*
19 Thijs Westerveld (UT) *Using generative probabilistic models for multimedia retrieval*
20 Madelon Evers (Nyenrode) *Learning from Design: facilitating multidisciplinary design teams*

**2005**

1 Floor Verdenius (UvA) *Methodological Aspects of Designing Induction-Based Applications*
2 Erik van der Werf (UM) *AI techniques for the game of Go*
3 Franc Grootjen (RUN) *A Pragmatic Approach to the Conceptualisation of Language*
4 Nirvana Meratnia (UT) *Towards Database Support for Moving Object data*
5 Gabriel Infante-Lopez (UvA) *Two-Level Probabilistic Grammars for Natural Language Parsing*
6 Pieter Spronck (UM) *Adaptive Game AI*
7 Flavius Frasincar (TUe) *Hypermedia Presentation Generation for Semantic Web Information Systems*
8 Richard Vdovjak (TUe) *A Model-driven Approach for Building Distributed Ontology-based Web Applications*
9 Jeen Broekstra (VUA) *Storage, Querying and Inferencing for Semantic Web Languages*
10 Anders Bouwer (UvA) *Explaining Behaviour: Using Qualitative Simulation in Interactive Learning Environments*
11 Elth Ogston (VUA) *Agent Based Matchmaking and Clustering: A Decentralized Approach to Search*
12 Csaba Boer (EUR) *Distributed Simulation in Industry*
13 Fred Hamburg (UL) *Een Computermodel voor het Ondersteunen van Euthanasiebeslissingen*
14 Borys Omelayenko (VUA) *Web-Service configuration on the Semantic Web: Exploring how semantics meets pragmatics*
15 Tibor Bosse (VUA) *Analysis of the Dynamics of Cognitive Processes*
16 Joris Graaumans (UU) *Usability of XML Query Languages*
17 Boris Shishkov (TUD) *Software Specification Based on Re-usable Business Components*
18 Danielle Sent (UU) *Test-selection strategies for probabilistic networks*
19 Michel van Dartel (UM) *Situated Representation*
20 Cristina Coteanu (UL) *Cyber Consumer Law, State of the Art and Perspectives*
21 Wijnand Derks (UT) *Improving Concurrency and Recovery in Database Systems by Exploiting Application Semantics*

**2006**

1 Samuil Angelov (TUe) *Foundations of B2B Electronic Contracting*
2 Cristina Chisalita (VUA) *Contextual issues in the design and use of information technology in organizations*
3 Noor Christoph (UvA) *The role of metacognitive skills in learning to solve problems*
4 Marta Sabou (VUA) *Building Web Service Ontologies*
5 Cees Pierik (UU) *Validation Techniques for Object-Oriented Proof Outlines*
6 Ziv Baida (VUA) *Software-aided Service Bundling: Intelligent Methods & Tools for Graphical Service Modeling*
7 Marko Smiljanic (UT) *XML schema matching: balancing efficiency and effectiveness by means of clustering*
8 Eelco Herder (UT) *Forward, Back and Home Again: Analyzing User Behavior on the Web*

9 Mohamed Wahdan (UM) *Automatic Formulation of the Auditor's Opinion*
10 Ronny Siebes (VUA) *Semantic Routing in Peer-to-Peer Systems*
11 Joeri van Ruth (UT) *Flattening Queries over Nested Data Types*
12 Bert Bongers (VUA) *Interactivation: Towards an e-cology of people, our technological environment, and the arts*
13 Henk-Jan Lebbink (UU) *Dialogue and Decision Games for Information Exchanging Agents*
14 Johan Hoorn (VUA) *Software Requirements: Update, Upgrade, Redesign - towards a Theory of Requirements Change*
15 Rainer Malik (UU) *CONAN: Text Mining in the Biomedical Domain*
16 Carsten Riggelsen (UU) *Approximation Methods for Efficient Learning of Bayesian Networks*
17 Stacey Nagata (UU) *User Assistance for Multitasking with Interruptions on a Mobile Device*
18 Valentin Zhizhkun (UvA) *Graph transformation for Natural Language Processing*
19 Birna van Riemsdijk (UU) *Cognitive Agent Programming: A Semantic Approach*
20 Marina Velikova (UvT) *Monotone models for prediction in data mining*
21 Bas van Gils (RUN) *Aptness on the Web*
22 Paul de Vrieze (RUN) *Fundaments of Adaptive Personalisation*
23 Ion Juvina (UU) *Development of Cognitive Model for Navigating on the Web*
24 Laura Hollink (VUA) *Semantic Annotation for Retrieval of Visual Resources*
25 Madalina Drugan (UU) *Conditional log-likelihood MDL and Evolutionary MCMC*
26 Vojkan Mihajlovic (UT) *Score Region Algebra: A Flexible Framework for Structured Information Retrieval*
27 Stefano Bocconi (CWI) *Vox Populi: generating video documentaries from semantically annotated media repositories*
28 Borkur Sigurbjornsson (UvA) *Focused Information Access using XML Element Retrieval*

**2007**

1 Kees Leune (UvT) *Access Control and Service-Oriented Architectures*
2 Wouter Teepe (RUG) *Reconciling Information Exchange and Confidentiality: A Formal Approach*
3 Peter Mika (VUA) *Social Networks and the Semantic Web*
4 Jurriaan van Diggelen (UU) *Achieving Semantic Interoperability in Multi-agent Systems: a dialogue-based approach*
5 Bart Schermer (UL) *Software Agents, Surveillance, and the Right to Privacy: a Legislative Framework for Agent-enabled Surveillance*
6 Gilad Mishne (UvA) *Applied Text Analytics for Blogs*
7 Natasa Jovanovic' (UT) *To Whom It May Concern: Addressee Identification in Face-to-Face Meetings*
8 Mark Hoogendoorn (VUA) *Modeling of Change in Multi-Agent Organizations*
9 David Mobach (VUA) *Agent-Based Mediated Service Negotiation*
10 Huib Aldewereld (UU) *Autonomy vs. Conformity: an Institutional Perspective on Norms and Protocols*
11 Natalia Stash (TUe) *Incorporating Cognitive/Learning Styles in a General-Purpose Adaptive Hypermedia System*
12 Marcel van Gerven (RUN) *Bayesian Networks for Clinical Decision Support: A Rational Approach to Dynamic Decision-Making under Uncertainty*
13 Rutger Rienks (UT) *Meetings in Smart Environments: Implications of Progressing Technology*
14 Niek Bergboer (UM) *Context-Based Image Analysis*
15 Joyca Lacroix (UM) *NIM: a Situated Computational Memory Model*
16 Davide Grossi (UU) *Designing Invisible Handcuffs. Formal investigations in Institutions and Organizations for Multi-agent Systems*
17 Theodore Charitos (UU) *Reasoning with Dynamic Networks in Practice*
18 Bart Orriens (UvT) *On the development an management of adaptive business collaborations*
19 David Levy (UM) *Intimate relationships with artificial partners*
20 Slinger Jansen (UU) *Customer Configuration Updating in a Software Supply Network*
21 Karianne Vermaas (UU) *Fast diffusion and broadening use: A research on residential adoption and usage of broadband internet in the Netherlands between 2001 and 2005*
22 Zlatko Zlatev (UT) *Goal-oriented design of value and process models from patterns*
23 Peter Barna (TUe) *Specification of Application Logic in Web Information Systems*
24 Georgina Ramírez Camps (CWI) *Structural Features in XML Retrieval*

25 Joost Schalken (VUA) *Empirical Investigations in Software Process Improvement*

**2008**

1 Katalin Boer-Sorbán (EUR) *Agent-Based Simulation of Financial Markets: A modular, continuous-time approach*
2 Alexei Sharpanskykh (VUA) *On Computer-Aided Methods for Modeling and Analysis of Organizations*
3 Vera Hollink (UvA) *Optimizing hierarchical menus: a usage-based approach*
4 Ander de Keijzer (UT) *Management of Uncertain Data: towards unattended integration*
5 Bela Mutschler (UT) *Modeling and simulating causal dependencies on process-aware information systems from a cost perspective*
6 Arjen Hommersom (RUN) *On the Application of Formal Methods to Clinical Guidelines, an Artificial Intelligence Perspective*
7 Peter van Rosmalen (OU) *Supporting the tutor in the design and support of adaptive e-learning*
8 Janneke Bolt (UU) *Bayesian Networks: Aspects of Approximate Inference*
9 Christof van Nimwegen (UU) *The paradox of the guided user: assistance can be counter-effective*
10 Wauter Bosma (UT) *Discourse oriented summarization*
11 Vera Kartseva (VUA) *Designing Controls for Network Organizations: A Value-Based Approach*
12 Jozsef Farkas (RUN) *A Semiotically Oriented Cognitive Model of Knowledge Representation*
13 Caterina Carraciolo (UvA) *Topic Driven Access to Scientific Handbooks*
14 Arthur van Bunningen (UT) *Context-Aware Querying: Better Answers with Less Effort*
15 Martijn van Otterlo (UT) *The Logic of Adaptive Behavior: Knowledge Representation and Algorithms for the Markov Decision Process Framework in First-Order Domains*
16 Henriette van Vugt (VUA) *Embodied agents from a user's perspective*
17 Martin Op 't Land (TUD) *Applying Architecture and Ontology to the Splitting and Allying of Enterprises*
18 Guido de Croon (UM) *Adaptive Active Vision*
19 Henning Rode (UT) *From Document to Entity Retrieval: Improving Precision and Performance of Focused Text Search*
20 Rex Arendsen (UvA) *Geen bericht, goed bericht. Een onderzoek naar de effecten van de introductie van elektronisch berichtenverkeer met de overheid op de administratieve lasten van bedrijven*
21 Krisztian Balog (UvA) *People Search in the Enterprise*
22 Henk Koning (UU) *Communication of IT-Architecture*
23 Stefan Visscher (UU) *Bayesian network models for the management of ventilator-associated pneumonia*
24 Zharko Aleksovski (VUA) *Using background knowledge in ontology matching*
25 Geert Jonker (UU) *Efficient and Equitable Exchange in Air Traffic Management Plan Repair using Spender-signed Currency*
26 Marijn Huijbregts (UT) *Segmentation, Diarization and Speech Transcription: Surprise Data Unraveled*
27 Hubert Vogten (OU) *Design and Implementation Strategies for IMS Learning Design*
28 Ildiko Flesch (RUN) *On the Use of Independence Relations in Bayesian Networks*
29 Dennis Reidsma (UT) *Annotations and Subjective Machines: Of Annotators, Embodied Agents, Users, and Other Humans*
30 Wouter van Atteveldt (VUA) *Semantic Network Analysis: Techniques for Extracting, Representing and Querying Media Content*
31 Loes Braun (UM) *Pro-Active Medical Information Retrieval*
32 Trung H. Bui (UT) *Toward Affective Dialogue Management using Partially Observable Markov Decision Processes*
33 Frank Terpstra (UvA) *Scientific Workflow Design: theoretical and practical issues*
34 Jeroen de Knijf (UU) *Studies in Frequent Tree Mining*
35 Ben Torben Nielsen (UvT) *Dendritic morphologies: function shapes structure*

**2009**

1 Rasa Jurgelenaite (RUN) *Symmetric Causal Independence Models*
2 Willem Robert van Hage (VUA) *Evaluating Ontology-Alignment Techniques*
3 Hans Stol (UvT) *A Framework for Evidence-based Policy Making Using IT*
4 Josephine Nabukenya (RUN) *Improving the Quality of Organisational Policy Making using Collaboration Engineering*

5 Sietse Overbeek (RUN) *Bridging Supply and Demand for Knowledge Intensive Tasks: Based on Knowledge, Cognition, and Quality*

6 Muhammad Subianto (UU) *Understanding Classification*

7 Ronald Poppe (UT) *Discriminative Vision-Based Recovery and Recognition of Human Motion*

8 Volker Nannen (VUA) *Evolutionary Agent-Based Policy Analysis in Dynamic Environments*

9 Benjamin Kanagwa (RUN) *Design, Discovery and Construction of Service-oriented Systems*

10 Jan Wielemaker (UvA) *Logic programming for knowledge-intensive interactive applications*

11 Alexander Boer (UvA) *Legal Theory, Sources of Law & the Semantic Web*

12 Peter Massuthe (TUE, Humboldt-Universitaet zu Berlin) *Operating Guidelines for Services*

13 Steven de Jong (UM) *Fairness in Multi-Agent Systems*

14 Maksym Korotkiy (VUA) *From ontology-enabled services to service-enabled ontologies (making ontologies work in e-science with ONTO-SOA)*

15 Rinke Hoekstra (UvA) *Ontology Representation: Design Patterns and Ontologies that Make Sense*

16 Fritz Reul (UvT) *New Architectures in Computer Chess*

17 Laurens van der Maaten (UvT) *Feature Extraction from Visual Data*

18 Fabian Groffen (CWI) *Armada, An Evolving Database System*

19 Valentin Robu (CWI) *Modeling Preferences, Strategic Reasoning and Collaboration in Agent-Mediated Electronic Markets*

20 Bob van der Vecht (UU) *Adjustable Autonomy: Controling Influences on Decision Making*

21 Stijn Vanderlooy (UM) *Ranking and Reliable Classification*

22 Pavel Serdyukov (UT) *Search For Expertise: Going beyond direct evidence*

23 Peter Hofgesang (VUA) *Modelling Web Usage in a Changing Environment*

24 Annerieke Heuvelink (VUA) *Cognitive Models for Training Simulations*

25 Alex van Ballegooij (CWI) *RAM: Array Database Management through Relational Mapping*

26 Fernando Koch (UU) *An Agent-Based Model for the Development of Intelligent Mobile Services*

27 Christian Glahn (OU) *Contextual Support of social Engagement and Reflection on the Web*

28 Sander Evers (UT) *Sensor Data Management with Probabilistic Models*

29 Stanislav Pokraev (UT) *Model-Driven Semantic Integration of Service-Oriented Applications*

30 Marcin Zukowski (CWI) *Balancing vectorized query execution with bandwidth-optimized storage*

31 Sofiya Katrenko (UvA) *A Closer Look at Learning Relations from Text*

32 Rik Farenhorst (VUA) *Architectural Knowledge Management: Supporting Architects and Auditors*

33 Khiet Truong (UT) *How Does Real Affect Affect Affect Recognition In Speech?*

34 Inge van de Weerd (UU) *Advancing in Software Product Management: An Incremental Method Engineering Approach*

35 Wouter Koelewijn (UL) *Privacy en Politiegegevens: Over geautomatiseerde normatieve informatie-uitwisseling*

36 Marco Kalz (OUN) *Placement Support for Learners in Learning Networks*

37 Hendrik Drachsler (OUN) *Navigation Support for Learners in Informal Learning Networks*

38 Riina Vuorikari (OU) *Tags and self-organisation: a metadata ecology for learning resources in a multilingual context*

39 Christian Stahl (TUE, Humboldt-Universitaet zu Berlin) *Service Substitution: A Behavioral Approach Based on Petri Nets*

40 Stephan Raaijmakers (UvT) *Multinomial Language Learning: Investigations into the Geometry of Language*

41 Igor Berezhnyy (UvT) *Digital Analysis of Paintings*

42 Toine Bogers (UvT) *Recommender Systems for Social Bookmarking*

43 Virginia Nunes Leal Franqueira (UT) *Finding Multi-step Attacks in Computer Networks using Heuristic Search and Mobile Ambients*

44 Roberto Santana Tapia (UT) *Assessing Business-IT Alignment in Networked Organizations*

45 Jilles Vreeken (UU) *Making Pattern Mining Useful*

46 Loredana Afanasiev (UvA) *Querying XML: Benchmarks and Recursion*

**2010**

1 Matthijs van Leeuwen (UU) *Patterns that Matter*
2 Ingo Wassink (UT) *Work flows in Life Science*
3 Joost Geurts (CWI) *A Document Engineering Model and Processing Framework for Multimedia documents*
4 Olga Kulyk (UT) *Do You Know What I Know? Situational Awareness of Co-located Teams in Multidisplay Environments*
5 Claudia Hauff (UT) *Predicting the Effectiveness of Queries and Retrieval Systems*
6 Sander Bakkes (UvT) *Rapid Adaptation of Video Game AI*
7 Wim Fikkert (UT) *Gesture interaction at a Distance*
8 Krzysztof Siewicz (UL) *Towards an Improved Regulatory Framework of Free Software. Protecting user freedoms in a world of software communities and eGovernments*
9 Hugo Kielman (UL) *A Politiele gegevensverwerking en Privacy, Naar een effectieve waarborging*
10 Rebecca Ong (UL) *Mobile Communication and Protection of Children*
11 Adriaan Ter Mors (TUD) *The world according to MARP: Multi-Agent Route Planning*
12 Susan van den Braak (UU) *Sensemaking software for crime analysis*
13 Gianluigi Folino (RUN) *High Performance Data Mining using Bio-inspired techniques*
14 Sander van Splunter (VUA) *Automated Web Service Reconfiguration*
15 Lianne Bodenstaff (UT) *Managing Dependency Relations in Inter-Organizational Models*
16 Sicco Verwer (TUD) *Efficient Identification of Timed Automata, theory and practice*
17 Spyros Kotoulas (VUA) *Scalable Discovery of Networked Resources: Algorithms, Infrastructure, Applications*
18 Charlotte Gerritsen (VUA) *Caught in the Act: Investigating Crime by Agent-Based Simulation*
19 Henriette Cramer (UvA) *People's Responses to Autonomous and Adaptive Systems*
20 Ivo Swartjes (UT) *Whose Story Is It Anyway? How Improv Informs Agency and Authorship of Emergent Narrative*
21 Harold van Heerde (UT) *Privacy-aware data management by means of data degradation*
22 Michiel Hildebrand (CWI) *End-user Support for Access to Heterogeneous Linked Data*
23 Bas Steunebrink (UU) *The Logical Structure of Emotions*
24 Zulfiqar Ali Memon (VUA) *Modelling Human-Awareness for Ambient Agents: A Human Mindreading Perspective*
25 Ying Zhang (CWI) *XRPC: Efficient Distributed Query Processing on Heterogeneous XQuery Engines*
26 Marten Voulon (UL) *Automatisch contracteren*
27 Arne Koopman (UU) *Characteristic Relational Patterns*
28 Stratos Idreos (CWI) *Database Cracking: Towards Auto-tuning Database Kernels*
29 Marieke van Erp (UvT) *Accessing Natural History: Discoveries in data cleaning, structuring, and retrieval*
30 Victor de Boer (UvA) *Ontology Enrichment from Heterogeneous Sources on the Web*
31 Marcel Hiel (UvT) *An Adaptive Service Oriented Architecture: Automatically solving Interoperability Problems*
32 Robin Aly (UT) *Modeling Representation Uncertainty in Concept-Based Multimedia Retrieval*
33 Teduh Dirgahayu (UT) *Interaction Design in Service Compositions*
34 Dolf Trieschnigg (UT) *Proof of Concept: Concept-based Biomedical Information Retrieval*
35 Jose Janssen (OU) *Paving the Way for Lifelong Learning: Facilitating competence development through a learning path specification*
36 Niels Lohmann (TUe) *Correctness of services and their composition*
37 Dirk Fahland (TUe) *From Scenarios to components*
38 Ghazanfar Farooq Siddiqui (VUA) *Integrative modeling of emotions in virtual agents*
39 Mark van Assem (VUA) *Converting and Integrating Vocabularies for the Semantic Web*
40 Guillaume Chaslot (UM) *Monte-Carlo Tree Search*

41 Sybren de Kinderen (VUA) *Needs-driven service bundling in a multi-supplier setting: the computational e3-service approach*
42 Peter van Kranenburg (UU) *A Computational Approach to Content-Based Retrieval of Folk Song Melodies*
43 Pieter Bellekens (TUe) *An Approach towards Context-sensitive and User-adapted Access to Heterogeneous Data Sources, Illustrated in the Television Domain*
44 Vasilios Andrikopoulos (UvT) *A theory and model for the evolution of software services*
45 Vincent Pijpers (VUA) *e3alignment: Exploring Inter-Organizational Business-ICT Alignment*
46 Chen Li (UT) *Mining Process Model Variants: Challenges, Techniques, Examples*
47 Jahn-Takeshi Saito (UM) *Solving difficult game positions*
48 Bouke Huurnink (UvA) *Search in Audiovisual Broadcast Archives*
49 Alia Khairia Amin (CWI) *Understanding and supporting information seeking tasks in multiple sources*
50 Peter-Paul van Maanen (VUA) *Adaptive Support for Human-Computer Teams: Exploring the Use of Cognitive Models of Trust and Attention*
51 Edgar Meij (UvA) *Combining Concepts and Language Models for Information Access*

**2011**

1 Botond Cseke (RUN) *Variational Algorithms for Bayesian Inference in Latent Gaussian Models*
2 Nick Tinnemeier (UU) *Organizing Agent Organizations. Syntax and Operational Semantics of an Organization-Oriented Programming Language*
3 Jan Martijn van der Werf (TUe) *Compositional Design and Verification of Component-Based Information Systems*
4 Hado van Hasselt (UU) *Insights in Reinforcement Learning: Formal analysis and empirical evaluation of temporal-difference*
5 Base van der Raadt (VUA) *Enterprise Architecture Coming of Age: Increasing the Performance of an Emerging Discipline*
6 Yiwen Wang (TUe) *Semantically-Enhanced Recommendations in Cultural Heritage*
7 Yujia Cao (UT) *Multimodal Information Presentation for High Load Human Computer Interaction*
8 Nieske Vergunst (UU) *BDI-based Generation of Robust Task-Oriented Dialogues*
9 Tim de Jong (OU) *Contextualised Mobile Media for Learning*
10 Bart Bogaert (UvT) *Cloud Content Contention*
11 Dhaval Vyas (UT) *Designing for Awareness: An Experience-focused HCI Perspective*
12 Carmen Bratosin (TUe) *Grid Architecture for Distributed Process Mining*
13 Xiaoyu Mao (UvT) *Airport under Control. Multiagent Scheduling for Airport Ground Handling*
14 Milan Lovric (EUR) *Behavioral Finance and Agent-Based Artificial Markets*
15 Marijn Koolen (UvA) *The Meaning of Structure: the Value of Link Evidence for Information Retrieval*
16 Maarten Schadd (UM) *Selective Search in Games of Different Complexity*
17 Jiyin He (UvA) *Exploring Topic Structure: Coherence, Diversity and Relatedness*
18 Mark Ponsen (UM) *Strategic Decision-Making in complex games*
19 Ellen Rusman (OU) *The Mind ' s Eye on Personal Profiles*
20 Qing Gu (VUA) *Guiding service-oriented software engineering: A view-based approach*
21 Linda Terlouw (TUD) *Modularization and Specification of Service-Oriented Systems*
22 Junte Zhang (UvA) *System Evaluation of Archival Description and Access*
23 Wouter Weerkamp (UvA) *Finding People and their Utterances in Social Media*
24 Herwin van Welbergen (UT) *Behavior Generation for Interpersonal Coordination with Virtual Humans On Specifying, Scheduling and Realizing Multimodal Virtual Human Behavior*
25 Syed Waqar ul Qounain Jaffry (VUA) *Analysis and Validation of Models for Trust Dynamics*
26 Matthijs Aart Pontier (VUA) *Virtual Agents for Human Communication: Emotion Regulation and Involvement-Distance Trade-Offs in Embodied Conversational Agents and Robots*
27 Aniel Bhulai (VUA) *Dynamic website optimization through autonomous management of design patterns*
28 Rianne Kaptein (UvA) *Effective Focused Retrieval by Exploiting Query Context and Document Structure*
29 Faisal Kamiran (TUe) *Discrimination-aware Classification*

30 Egon van den Broek (UT) *Affective Signal Processing (ASP): Unraveling the mystery of emotions*
31 Ludo Waltman (EUR) *Computational and Game-Theoretic Approaches for Modeling Bounded Rationality*
32 Nees-Jan van Eck (EUR) *Methodological Advances in Bibliometric Mapping of Science*
33 Tom van der Weide (UU) *Arguing to Motivate Decisions*
34 Paolo Turrini (UU) *Strategic Reasoning in Interdependence: Logical and Game-theoretical Investigations*
35 Maaike Harbers (UU) *Explaining Agent Behavior in Virtual Training*
36 Erik van der Spek (UU) *Experiments in serious game design: a cognitive approach*
37 Adriana Burlutiu (RUN) *Machine Learning for Pairwise Data, Applications for Preference Learning and Supervised Network Inference*
38 Nyree Lemmens (UM) *Bee-inspired Distributed Optimization*
39 Joost Westra (UU) *Organizing Adaptation using Agents in Serious Games*
40 Viktor Clerc (VUA) *Architectural Knowledge Management in Global Software Development*
41 Luan Ibraimi (UT) *Cryptographically Enforced Distributed Data Access Control*
42 Michal Sindlar (UU) *Explaining Behavior through Mental State Attribution*
43 Henk van der Schuur (UU) *Process Improvement through Software Operation Knowledge*
44 Boris Reuderink (UT) *Robust Brain-Computer Interfaces*
45 Herman Stehouwer (UvT) *Statistical Language Models for Alternative Sequence Selection*
46 Beibei Hu (TUD) *Towards Contextualized Information Delivery: A Rule-based Architecture for the Domain of Mobile Police Work*
47 Azizi Bin Ab Aziz (VUA) *Exploring Computational Models for Intelligent Support of Persons with Depression*
48 Mark Ter Maat (UT) *Response Selection and Turn-taking for a Sensitive Artificial Listening Agent*
49 Andreea Niculescu (UT) *Conversational interfaces for task-oriented spoken dialogues: design aspects influencing interaction quality*

**2012**

1 Terry Kakeeto (UvT) *Relationship Marketing for SMEs in Uganda*
2 Muhammad Umair (VUA) *Adaptivity, emotion, and Rationality in Human and Ambient Agent Models*
3 Adam Vanya (VUA) *Supporting Architecture Evolution by Mining Software Repositories*
4 Jurriaan Souer (UU) *Development of Content Management System-based Web Applications*
5 Marijn Plomp (UU) *Maturing Interorganisational Information Systems*
6 Wolfgang Reinhardt (OU) *Awareness Support for Knowledge Workers in Research Networks*
7 Rianne van Lambalgen (VUA) *When the Going Gets Tough: Exploring Agent-based Models of Human Performance under Demanding Conditions*
8 Gerben de Vries (UvA) *Kernel Methods for Vessel Trajectories*
9 Ricardo Neisse (UT) *Trust and Privacy Management Support for Context-Aware Service Platforms*
10 David Smits (TUe) *Towards a Generic Distributed Adaptive Hypermedia Environment*
11 J. C. B. Rantham Prabhakara (TUe) *Process Mining in the Large: Preprocessing, Discovery, and Diagnostics*
12 Kees van der Sluijs (TUe) *Model Driven Design and Data Integration in Semantic Web Information Systems*
13 Suleman Shahid (UvT) *Fun and Face: Exploring non-verbal expressions of emotion during playful interactions*
14 Evgeny Knutov (TUe) *Generic Adaptation Framework for Unifying Adaptive Web-based Systems*
15 Natalie van der Wal (VUA) *Social Agents. Agent-Based Modelling of Integrated Internal and Social Dynamics of Cognitive and Affective Processes*
16 Fiemke Both (VUA) *Helping people by understanding them: Ambient Agents supporting task execution and depression treatment*
17 Amal Elgammal (UvT) *Towards a Comprehensive Framework for Business Process Compliance*
18 Eltjo Poort (VUA) *Improving Solution Architecting Practices*
19 Helen Schonenberg (TUe) *What's Next? Operational Support for Business Process Execution*

20 Ali Bahramisharif (RUN) *Covert Visual Spatial Attention, a Robust Paradigm for Brain-Computer Interfacing*
21 Roberto Cornacchia (TUD) *Querying Sparse Matrices for Information Retrieval*
22 Thijs Vis (UvT) *Intelligence, politie en veiligheidsdienst: verenigbare grootheden?*
23 Christian Muehl (UT) *Toward Affective Brain-Computer Interfaces: Exploring the Neurophysiology of Affect during Human Media Interaction*
24 Laurens van der Werff (UT) *Evaluation of Noisy Transcripts for Spoken Document Retrieval*
25 Silja Eckartz (UT) *Managing the Business Case Development in Inter-Organizational IT Projects: A Methodology and its Application*
26 Emile de Maat (UvA) *Making Sense of Legal Text*
27 Hayrettin Gurkok (UT) *Mind the Sheep! User Experience Evaluation & Brain-Computer Interface Games*
28 Nancy Pascall (UvT) *Engendering Technology Empowering Women*
29 Almer Tigelaar (UT) *Peer-to-Peer Information Retrieval*
30 Alina Pommeranz (TUD) *Designing Human-Centered Systems for Reflective Decision Making*
31 Emily Bagarukayo (RUN) *A Learning by Construction Approach for Higher Order Cognitive Skills Improvement, Building Capacity and Infrastructure*
32 Wietske Visser (TUD) *Qualitative multi-criteria preference representation and reasoning*
33 Rory Sie (OUN) *Coalitions in Cooperation Networks (COCOON)*
34 Pavol Jancura (RUN) *Evolutionary analysis in PPI networks and applications*
35 Evert Haasdijk (VUA) *Never Too Old To Learn: On-line Evolution of Controllers in Swarm- and Modular Robotics*
36 Denis Ssebugwawo (RUN) *Analysis and Evaluation of Collaborative Modeling Processes*
37 Agnes Nakakawa (RUN) *A Collaboration Process for Enterprise Architecture Creation*
38 Selmar Smit (VUA) *Parameter Tuning and Scientific Testing in Evolutionary Algorithms*
39 Hassan Fatemi (UT) *Risk-aware design of value and coordination networks*
40 Agus Gunawan (UvT) *Information Access for SMEs in Indonesia*
41 Sebastian Kelle (OU) *Game Design Patterns for Learning*
42 Dominique Verpoorten (OU) *Reflection Amplifiers in self-regulated Learning*
43 Anna Tordai (VUA) *On Combining Alignment Techniques*
44 Benedikt Kratz (UvT) *A Model and Language for Business-aware Transactions*
45 Simon Carter (UvA) *Exploration and Exploitation of Multilingual Data for Statistical Machine Translation*
46 Manos Tsagkias (UvA) *Mining Social Media: Tracking Content and Predicting Behavior*
47 Jorn Bakker (TUe) *Handling Abrupt Changes in Evolving Time-series Data*
48 Michael Kaisers (UM) *Learning against Learning: Evolutionary dynamics of reinforcement learning algorithms in strategic interactions*
49 Steven van Kervel (TUD) *Ontologogy driven Enterprise Information Systems Engineering*
50 Jeroen de Jong (TUD) *Heuristics in Dynamic Sceduling: a practical framework with a case study in elevator dispatching*

**2013**

1 Viorel Milea (EUR) *News Analytics for Financial Decision Support*
2 Erietta Liarou (CWI) *MonetDB/DataCell: Leveraging the Column-store Database Technology for Efficient and Scalable Stream Processing*
3 Szymon Klarman (VUA) *Reasoning with Contexts in Description Logics*
4 Chetan Yadati (TUD) *Coordinating autonomous planning and scheduling*
5 Dulce Pumareja (UT) *Groupware Requirements Evolutions Patterns*
6 Romulo Goncalves (CWI) *The Data Cyclotron: Juggling Data and Queries for a Data Warehouse Audience*
7 Giel van Lankveld (UvT) *Quantifying Individual Player Differences*
8 Robbert-Jan Merk (VUA) *Making enemies: cognitive modeling for opponent agents in fighter pilot simulators*
9 Fabio Gori (RUN) *Metagenomic Data Analysis: Computational Methods and Applications*
10 Jeewanie Jayasinghe Arachchige (UvT) *A Unified Modeling Framework for Service Design*

11 Evangelos Pournaras (TUD) *Multi-level Reconfigurable Self-organization in Overlay Services*
12 Marian Razavian (VUA) *Knowledge-driven Migration to Services*
13 Mohammad Safiri (UT) *Service Tailoring: User-centric creation of integrated IT-based homecare services to support independent living of elderly*
14 Jafar Tanha (UvA) *Ensemble Approaches to Semi-Supervised Learning Learning*
15 Daniel Hennes (UM) *Multiagent Learning: Dynamic Games and Applications*
16 Eric Kok (UU) *Exploring the practical benefits of argumentation in multi-agent deliberation*
17 Koen Kok (VUA) *The PowerMatcher: Smart Coordination for the Smart Electricity Grid*
18 Jeroen Janssens (UvT) *Outlier Selection and One-Class Classification*
19 Renze Steenhuizen (TUD) *Coordinated Multi-Agent Planning and Scheduling*
20 Katja Hofmann (UvA) *Fast and Reliable Online Learning to Rank for Information Retrieval*
21 Sander Wubben (UvT) *Text-to-text generation by monolingual machine translation*
22 Tom Claassen (RUN) *Causal Discovery and Logic*
23 Patricio de Alencar Silva (UvT) *Value Activity Monitoring*
24 Haitham Bou Ammar (UM) *Automated Transfer in Reinforcement Learning*
25 Agnieszka Anna Latoszek-Berendsen (UM) *Intention-based Decision Support. A new way of representing and implementing clinical guidelines in a Decision Support System*
26 Alireza Zarghami (UT) *Architectural Support for Dynamic Homecare Service Provisioning*
27 Mohammad Huq (UT) *Inference-based Framework Managing Data Provenance*
28 Frans van der Sluis (UT) *When Complexity becomes Interesting: An Inquiry into the Information eXperience*
29 Iwan de Kok (UT) *Listening Heads*
30 Joyce Nakatumba (TUe) *Resource-Aware Business Process Management: Analysis and Support*
31 Dinh Khoa Nguyen (UvT) *Blueprint Model and Language for Engineering Cloud Applications*
32 Kamakshi Rajagopal (OUN) *Networking For Learning: The role of Networking in a Lifelong Learner's Professional Development*
33 Qi Gao (TUD) *User Modeling and Personalization in the Microblogging Sphere*
34 Kien Tjin-Kam-Jet (UT) *Distributed Deep Web Search*
35 Abdallah El Ali (UvA) *Minimal Mobile Human Computer Interaction*
36 Than Lam Hoang (TUe) *Pattern Mining in Data Streams*
37 Dirk Börner (OUN) *Ambient Learning Displays*
38 Eelco den Heijer (VUA) *Autonomous Evolutionary Art*
39 Joop de Jong (TUD) *A Method for Enterprise Ontology based Design of Enterprise Information Systems*
40 Pim Nijssen (UM) *Monte-Carlo Tree Search for Multi-Player Games*
41 Jochem Liem (UvA) *Supporting the Conceptual Modelling of Dynamic Systems: A Knowledge Engineering Perspective on Qualitative Reasoning*
42 Léon Planken (TUD) *Algorithms for Simple Temporal Reasoning*
43 Marc Bron (UvA) *Exploration and Contextualization through Interaction and Concepts*

**2014**

1 Nicola Barile (UU) *Studies in Learning Monotone Models from Data*
2 Fiona Tuliyano (RUN) *Combining System Dynamics with a Domain Modeling Method*
3 Sergio Raul Duarte Torres (UT) *Information Retrieval for Children: Search Behavior and Solutions*
4 Hanna Jochmann-Mannak (UT) *Websites for children: search strategies and interface design - Three studies on children's search performance and evaluation*
5 Jurriaan van Reijsen (UU) *Knowledge Perspectives on Advancing Dynamic Capability*
6 Damian Tamburri (VUA) *Supporting Networked Software Development*
7 Arya Adriansyah (TUe) *Aligning Observed and Modeled Behavior*
8 Samur Araujo (TUD) *Data Integration over Distributed and Heterogeneous Data Endpoints*

9 Philip Jackson (UvT) *Toward Human-Level Artificial Intelligence: Representation and Computation of Meaning in Natural Language*
10 Ivan Salvador Razo Zapata (VUA) *Service Value Networks*
11 Janneke van der Zwaan (TUD) *An Empathic Virtual Buddy for Social Support*
12 Willem van Willigen (VUA) *Look Ma, No Hands: Aspects of Autonomous Vehicle Control*
13 Arlette van Wissen (VUA) *Agent-Based Support for Behavior Change: Models and Applications in Health and Safety Domains*
14 Yangyang Shi (TUD) *Language Models With Meta-information*
15 Natalya Mogles (VUA) *Agent-Based Analysis and Support of Human Functioning in Complex Socio-Technical Systems: Applications in Safety and Healthcare*
16 Krystyna Milian (VUA) *Supporting trial recruitment and design by automatically interpreting eligibility criteria*
17 Kathrin Dentler (VUA) *Computing healthcare quality indicators automatically: Secondary Use of Patient Data and Semantic Interoperability*
18 Mattijs Ghijsen (UvA) *Methods and Models for the Design and Study of Dynamic Agent Organizations*
19 Vinicius Ramos (TUe) *Adaptive Hypermedia Courses: Qualitative and Quantitative Evaluation and Tool Support*
20 Mena Habib (UT) *Named Entity Extraction and Disambiguation for Informal Text: The Missing Link*
21 Kassidy Clark (TUD) *Negotiation and Monitoring in Open Environments*
22 Marieke Peeters (UU) *Personalized Educational Games: Developing agent-supported scenario-based training*
23 Eleftherios Sidirourgos (UvA/CWI) *Space Efficient Indexes for the Big Data Era*
24 Davide Ceolin (VUA) *Trusting Semi-structured Web Data*
25 Martijn Lappenschaar (RUN) *New network models for the analysis of disease interaction*
26 Tim Baarslag (TUD) *What to Bid and When to Stop*
27 Rui Jorge Almeida (EUR) *Conditional Density Models Integrating Fuzzy and Probabilistic Representations of Uncertainty*
28 Anna Chmielowiec (VUA) *Decentralized k-Clique Matching*
29 Jaap Kabbedijk (UU) *Variability in Multi-Tenant Enterprise Software*
30 Peter de Cock (UvT) *Anticipating Criminal Behaviour*
31 Leo van Moergestel (UU) *Agent Technology in Agile Multiparallel Manufacturing and Product Support*
32 Naser Ayat (UvA) *On Entity Resolution in Probabilistic Data*
33 Tesfa Tegegne (RUN) *Service Discovery in eHealth*
34 Christina Manteli (VUA) *The Effect of Governance in Global Software Development: Analyzing Transactive Memory Systems*
35 Joost van Ooijen (UU) *Cognitive Agents in Virtual Worlds: A Middleware Design Approach*
36 Joos Buijs (TUe) *Flexible Evolutionary Algorithms for Mining Structured Process Models*
37 Maral Dadvar (UT) *Experts and Machines United Against Cyberbullying*
38 Danny Plass-Oude Bos (UT) *Making brain-computer interfaces better: improving usability through post-processing*
39 Jasmina Maric (UvT) *Web Communities, Immigration, and Social Capital*
40 Walter Omona (RUN) *A Framework for Knowledge Management Using ICT in Higher Education*
41 Frederic Hogenboom (EUR) *Automated Detection of Financial Events in News Text*
42 Carsten Eijckhof (CWI/TUD) *Contextual Multidimensional Relevance Models*
43 Kevin Vlaanderen (UU) *Supporting Process Improvement using Method Increments*
44 Paulien Meesters (UvT) *Intelligent Blauw: Intelligence-gestuurde politiezorg in gebiedsgebonden eenheden*
45 Birgit Schmitz (OUN) *Mobile Games for Learning: A Pattern-Based Approach*
46 Ke Tao (TUD) *Social Web Data Analytics: Relevance, Redundancy, Diversity*
47 Shangsong Liang (UvA) *Fusion and Diversification in Information Retrieval*

**2015**

1 Niels Netten (UvA) *Machine Learning for Relevance of Information in Crisis Response*

2 Faiza Bukhsh (UvT) *Smart auditing: Innovative Compliance Checking in Customs Controls*
3 Twan van Laarhoven (RUN) *Machine learning for network data*
4 Howard Spoelstra (OUN) *Collaborations in Open Learning Environments*
5 Christoph Bösch (UT) *Cryptographically Enforced Search Pattern Hiding*
6 Farideh Heidari (TUD) *Business Process Quality Computation: Computing Non-Functional Requirements to Improve Business Processes*
7 Maria-Hendrike Peetz (UvA) *Time-Aware Online Reputation Analysis*
8 Jie Jiang (TUD) *Organizational Compliance: An agent-based model for designing and evaluating organizational interactions*
9 Randy Klaassen (UT) *HCI Perspectives on Behavior Change Support Systems*
10 Henry Hermans (OUN) *OpenU: design of an integrated system to support lifelong learning*
11 Yongming Luo (TUe) *Designing algorithms for big graph datasets: A study of computing bisimulation and joins*
12 Julie M. Birkholz (VUA) *Modi Operandi of Social Network Dynamics: The Effect of Context on Scientific Collaboration Networks*
13 Giuseppe Procaccianti (VUA) *Energy-Efficient Software*
14 Bart van Straalen (UT) *A cognitive approach to modeling bad news conversations*
15 Klaas Andries de Graaf (VUA) *Ontology-based Software Architecture Documentation*
16 Changyun Wei (UT) *Cognitive Coordination for Cooperative Multi-Robot Teamwork*
17 André van Cleeff (UT) *Physical and Digital Security Mechanisms: Properties, Combinations and Trade-offs*
18 Holger Pirk (CWI) *Waste Not, Want Not!: Managing Relational Data in Asymmetric Memories*
19 Bernardo Tabuenca (OUN) *Ubiquitous Technology for Lifelong Learners*
20 Loïs Vanhée (UU) *Using Culture and Values to Support Flexible Coordination*
21 Sibren Fetter (OUN) *Using Peer-Support to Expand and Stabilize Online Learning*
22 Zhemin Zhu (UT) *Co-occurrence Rate Networks*
23 Luit Gazendam (VUA) *Cataloguer Support in Cultural Heritage*
24 Richard Berendsen (UvA) *Finding People, Papers, and Posts: Vertical Search Algorithms and Evaluation*
25 Steven Woudenberg (UU) *Bayesian Tools for Early Disease Detection*
26 Alexander Hogenboom (EUR) *Sentiment Analysis of Text Guided by Semantics and Structure*
27 Sándor Héman (CWI) *Updating compressed column-stores*
28 Janet Bagorogoza (TiU) *Knowledge Management and High Performance: The Uganda Financial Institutions Model for HPO*
29 Hendrik Baier (UM) *Monte-Carlo Tree Search Enhancements for One-Player and Two-Player Domains*
30 Kiavash Bahreini (OUN) *Real-time Multimodal Emotion Recognition in E-Learning*
31 Yakup Koç (TUD) *On Robustness of Power Grids*
32 Jerome Gard (UL) *Corporate Venture Management in SMEs*
33 Frederik Schadd (UM) *Ontology Mapping with Auxiliary Resources*
34 Victor de Graaff (UT) *Geosocial Recommender Systems*
35 Junchao Xu (TUD) *Affective Body Language of Humanoid Robots: Perception and Effects in Human Robot Interaction*

**2016**

1 Syed Saiden Abbas (RUN) *Recognition of Shapes by Humans and Machines*
2 Michiel Christiaan Meulendijk (UU) *Optimizing medication reviews through decision support: prescribing a better pill to swallow*
3 Maya Sappelli (RUN) *Knowledge Work in Context: User Centered Knowledge Worker Support*
4 Laurens Rietveld (VUA) *Publishing and Consuming Linked Data*
5 Evgeny Sherkhonov (UvA) *Expanded Acyclic Queries: Containment and an Application in Explaining Missing Answers*
6 Michel Wilson (TUD) *Robust scheduling in an uncertain environment*
7 Jeroen de Man (VUA) *Measuring and modeling negative emotions for virtual training*
8 Matje van de Camp (TiU) *A Link to the Past: Constructing Historical Social Networks from Unstructured Data*
9 Archana Nottamkandath (VUA) *Trusting Crowdsourced Information on Cultural Artefacts*

10 George Karafotias (VUA) *Parameter Control for Evolutionary Algorithms*
11 Anne Schuth (UvA) *Search Engines that Learn from Their Users*
12 Max Knobbout (UU) *Logics for Modelling and Verifying Normative Multi-Agent Systems*
13 Nana Baah Gyan (VU) *The Web, Speech Technologies and Rural Development in West Africa – An ICT4D Approach*
14 Ravi Khadka (UU) *Revisiting Legacy Software System Modernization*
15 Steffen Michels (RUN) *Hybrid Probabilistic Logics – Theoretical Aspects, Algorithms and Experiments*
16 Guangliang Li (UVA) *Socially Intelligent Autonomous Agents that Learn from Human Reward*
17 Berend Weel (VU) *Towards Embodied Evolution of Robot Organisms*
18 Albert Meroño Peñuela (VU) *Refining Statistical Data on the Web*
19 Julia Efremova (Tu/e) *Mining Social Structures from Genealogical Data*
20 Daan Odijk (UVA) *Context & Semantics in News & Web Search*
21 Alejandro Moreno Célleri (UT) *From Traditional to Interactive Playspaces: Automatic Analysis of Player Behavior in the Interactive Tag Playground*
22 Grace Lewis (VU) *Software Architecture Strategies for Cyber-Foraging Systems*
23 Fei Cai (UVA) *Query Auto Completion in Information Retrieval*
24 Brend Wanders (UT) *Repurposing and Probabilistic Integration of Data; An Iterative and data model independent approach*
25 Julia Kiseleva (TU/e) *Using Contextual Information to Understand Searching and Browsing Behavior*