Towards a Feature-Rich Data Set for Personalized Access to Long-Tail Content

Christin Seifert
University of Passau
Innstraße 41
Passau, Germany
christin.seifert@uni-passau.de

Jörg Schlötterer
University of Passau
Innstraße 41
Passau, Germany
joerg.schloetterer@uni-passau.de

Michael Granitzer
University of Passau
Innstraße 41
Passau, Germany
michael.granitzer@uni-passau.de

ABSTRACT

Personalized data access has become one of the core challenges for intelligent information access, especially for non-mainstream long-tail content, as can be found in digital libraries. One of the main reasons that personalization remains a difficult task is the lack of standardized test corpora. In this paper we provide a comprehensive analysis of feature requirements for personalization together with a data collection tool for generating user models and collecting data for personalization of search and recommender system optimization in the long-tail. Based on the feature analysis, we provide a feature-rich publicly available data set, covering web content consumption and creation tasks. Our data set contains user models for eight users, including performed tasks, relevant topics for each task, relevance ratings, and relations between focus text and search queries. Altogether, the data set consists of 217 tasks, 4562 queries and over 15.000 ratings. On this data we perform automatic query prediction from web page content, achieving an accuracy of 89% using term identity, capitalization and part-of-speech tags as features. The results of the feature analysis can serve as guideline for feature collection for long-tail content personalization, and the provided data set as a gold standard for learning and evaluation of user models as well as for optimizing recommender or search engines for long-tail domains.

Categories and Subject Descriptors

H.3.3 [**Information Systems**]: Information Search and Retrieval—*Search process*; H.3.4 [**Information Systems Applications**]: Systems and Software—*User profiles and alert services*

General Terms

EXPERIMENTATION, MEASUREMENT

SAC'15 April 13-17, 2015, Salamanca, Spain.
Copyright is held by the owner/author(s). Publication rights licensed to ACM.
ACM 978-1-4503-3196-8/15/04 ...$15.00.
http://dx.doi.org/10.1145/2695664.2695671.

Keywords

Personalization, Long-Tail Content, Gold Standard

1. INTRODUCTION

Long-tail sources in the Web [2] consist of content which gets considerably less visits than the information hubs like Google, Facebook and BBC-News. Examples for such content is educational, scientific or cultural material, which remains often buried in deep web portals like libraries or cultural organizations. Access to those resources is available only to users knowing the specialized repositories and for those having the ability to utilize non Google-like search interfaces. Personalization plays an important role for retrieving long-tail content from these specialized repositories. Without knowledge about the user and her current information need, a magnitude of potentially relevant resources exists and it remains impossible for an automatic mechanism to select the proper one. However, developing appropriate methods for personalized access to long-tail content is challenging due to the lack of proper test data sets.

In this paper we present an approach for test data acquisition for personalized access to long-tail content. The data acquisition is motivated by a user-centric information seeking model that provides different viewpoints on the user's context. Based on this model we derive a feature set, capable of testing different personalization strategies for long-tail content in a browser-based setting. As our main contribution, we present a publicly available, feature-rich data set collected from 8 users with 170 hours of user interaction, encompassing about 4,560 queries and over 15,000 ratings. In order to demonstrate the value of this data set, we provide results for automatically learning user queries utilizing conditional random fields. The prototype for data acquisition, a screencast of a user, performing related tasks and the data set are available online[1].

The rest of the paper is structured as follows: In section 2 we introduce the conceptual model for data collection. Section 3 provides a feature analysis for personalization strategies based on the information retrieval model of Fuhr [5]. Section 4 presents our methodology for data collection. Section 5 provides a comprehensive analysis of the collected data. A preliminary study for automatic query generation, shows one potential application scenario and demonstrates the value of the data set (section 6). Related work is reviewed in section 7 and section 8 concludes our work.

[1] `purl.org/eexcess/datasets/umlt`

2. AN INFORMATION SEEKING MODEL FOR TEST DATA ACQUISITION

Before gathering test data we need a suitable information seeking model that provides a rational for the information seeking tasks to be supported. Such a human centric model has been introduced by Marchionini & White [15] and consists of seven steps carried out by an information seeker. First, a need for information has to be *recognized*, and the challenge for satisfying this need has to be *accepted*. Then, the problem is *formulated* in the user's mind, meaning that important keywords and types of resources that would satisfy the information need (e.g. academic papers, movies) are identified. Finally, the information need is *expressed*, for example as query in a search interface. *Examination* of results is potentially followed by *reformulation* of the information need, and eventually the cycle is closed by the actual *usage* of results. The process is illustrated in figure 1.

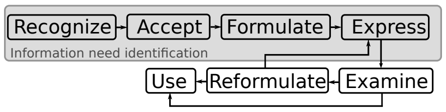


Figure 1: User-based information seeking model of [15]

If the first four steps, *recognition* to *expression*, are performed manually, the model reflects traditional search. Fully automatic detection and expression of the information need, however, resembles just-in-time-retrieval [21]. To achieve an accurate detection of the information need, i.e., to fully and satisfactory automate steps 1 to 4, the user's context has to be observed and turned into an appropriate expression of her information need. This expression, which may either be represented explicitely or implicitly, is then used to personalize the retrieval or recommendation results.

There are three paradigms for personalization in search: (i) a tight coupling between the personalization and the retrieval process, (ii) personalization as a pre-processing, and (iii) personalization as a post-processing step [17]. Similar paradigms have also been identified for recommender systems [1]. While the first approach requires access to search engine internals and considerable manual effort from information retrieval experts, the latter two approaches can be used in scenarios where the search system is seen as a black-box. Black-box scenarios may be caused by no direct access to search engine internals, by the necessity to federate the search [24] or by privacy considerations, where the search system is deemed untrustworthy and may not be allowed to receive the full user context information [20, 13]. Especially, the necessity for federated search is prominent for long-tail content, which is distributed across repositories, such as digital libraries or local aggregators for digital collections.

The focus of this paper is personalization of long-tail content, and therefore we further investigate personalization as pre- and personalization as post-processing to derive the features for our test data collection.

3. FEATURE REQUIREMENTS FOR DATA COLLECTION

In order to identify the features to collect, we employ and extend the information retrieval model by Fuhr [5] shown in figure 2. In the basic model the set of documents $\underline{D}$ is

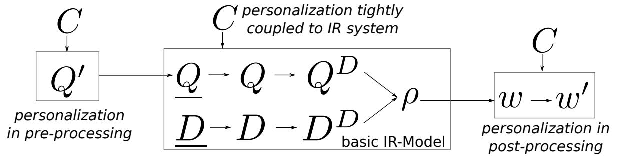

C
C personalization tightly coupled to IR system
C
Q'
$\underline{Q} \rightarrow Q \rightarrow Q^D$
$\underline{D} \rightarrow D \rightarrow D^D$
ρ
$w \rightarrow w'$
personalization in pre-processing
basic IR-Model
personalization in post-processing

Figure 2: Principle personalization approaches. Extended IR Model of Fuhr [5].

converted into a set of document representations D (document preprocessing), and the latter is then indexed to yield the set of document descriptions D^D . The same procedure is applied to the query set $\underline{Q}$, yielding a query description set Q^D. The retrieval function ρ then assigns a weight or score w to each document-query pair, i.e. $\rho : Q^D \times D^D \rightarrow \mathbb{R}$; $\rho(q^D, d^D) = w$. In our conceptual setting (federated search, search engines as black box) neither the process to generate the query description Q^D nor the retrieval function ρ can be accessed and modified for personalization. Thus, we do not discuss the case where the user context C is used within the retrieval system itself (tight coupling).

3.1 Personalization as Pre-Processing

Pre-processing methods for personalization aim to modify the initial user query Q' into a query $\underline{Q}$ that better reflects the information need w.r.t. the vocabulary in the document collection. Query refinement strategies can be categorized into global and local methods [14]. Global methods are applied independently of the specific query and search results, and basically refer to query rewriting based on user interests or related topics (as e.g. in [9]). Local methods change a query based on the initially retrieved results for this query. The Rocchio method for relevance feedback requires positive and/or negative feedback on retrieved results and works for queries with similar information need by adjusting the term weights of the query [22]. Also probabilistic relevance feedback methods, for instance based on the Naive Bayes classifier require positive feedback on results. According to [14] feedback for at least 5 items in the result set is advisable. Newer approaches incorporate implicit feedback, like the results the user has viewed and the time the user spent viewing a particular result. It is to note, that approaches that reweight the query terms require a search system capable of processing queries with weighted terms.

As conclusion, to collect features for the above mentioned methods, we need to collect the query, positive and negative feedback (for at least 5 items), result click data (document id and time spent), and user interests or relevant topics.

3.2 Personalization as Post-Processing

Post-processing methods for personalization aim to adapt the score w for a document-query pair to a new score w', such that in the result list, sorted by w', documents that satisfy the user's information need are higher-ranked. This can be evaluated by various measures, e.g., precision or mean reciprocal rank. Click-history based re-ranking [4] ranks the documents for user u based on previous clicks on documents for the same query q. In detail, the final score is calculated as $w'(q,d,u) = \frac{clicks(q,d,u)}{clicks(q,*,u)+\beta}$ with $clicks(q,d,u)$ being the number of clicks for query q, β a smoothing factor, d a document, u a user and $*$ a wildcard for all documents. For methods that re-rank the queries based on user interests [19], a vector representation of the user interests is compared with a vector representation of the retrieved documents, assigning a higher score to documents similar to user interests. The user interests can either be given manually, extracted from previously viewed documents or the browsing history.

As a conclusion, to collect features for the post-processing methods, we need to collect queries, clicks on result documents, and the browsing history.

3.3 Features to Collect

For the above mentioned personalization approaches we need to collect the user *query*, *positive and negative feedback* (for at least 5 items), result *click data* (document id and time spent) and *browsing history*. Additionally, we collected *topics* users manually assigned to a web page, based on its contents for assessing the usefulness of this information for query modification or as feature for content-based filtering. In order to provide not only relevant additional information, but also to decide when this information is of interest, we collected a binary feature indicating *if the user would require resources* and a label for the performed *task* (e.g. "watching funny videos on YouTube"). The latter was either selected from a predefined set or input as free form text. To elicit the requirements for cross-language result suggestions, we asked for the *preferred language of results*.

Further, we collected mouse clicks on the web page and text selections as strong cues for predicting the focus area of human readers [8]. With this information we want to assess how much and which of the information need can be specified via that focus area and what has to be inferred from additional context information (e.g. browsing history, topics, external sources, such as wordnet). Finally, by means of a questionnaire, we assess the *perceived sensitivity* and the *disclosure level* w.r.t. privacy considerations for the above mentioned features. We measure perceived sensitivity using a 5-point Likert scale ranging from highly sensitive to not sensitive at all. Disclosure level assesses the willingness to provide a specific piece of information for either "perfect results" , "improvement in result quality" or "never, regardless of the potential improvements".

Table 1 provides a summary of all features.

4. ACQUISITION METHODOLOGY

In order to collect the previously identified features, we developed a web browser extension and defined tasks for users to solve by using this extension. Users also were asked to fill out a questionnaire to collect privacy related features and demographic data.

4.1 Tasks

We distinguish between two basic web usage scenarios. The first scenario summarizes all web tasks with an intrinsic information need, i.e. where provision of additional relevant content is desirable. Such tasks are the focus of our interest. An example may be "researching on a topic in online sources". The second usage scenario contains web tasks, where presumably no additional resources are needed or would disturb the user, we call these tasks "background tasks", e.g., the task "home banking".

We further distinguish the tasks with information need based on the type of usage of the resources, as introduced in [7]. Specifically, we subsume tasks where content is merely consumed for personal use as *content consumption* tasks, and tasks that integrate the results in order to create and distribute new content as *content creation* tasks.

Table 1: Overview of collected features. E refers to data explicitly stated by users either via a user interface control (UI) or a questionnaire (Q), I to automatic recording of user behavior.

Feature	**Type**
mouse clicks (+target)	I
textual input	I
browsing history	I
browser profile (plugins, ...)	I
User and Task	
predefined task name	E UI
custom task name	E UI
task start-time	E UI
task end-time	E UI
level of expertise	E UI
topics relevant to task	E UI
demographic data	E Q
Resources	
recommendations desirable?	E UI
queries	E UI
preferred language of results	E UI
rating of results (binary)	E UI
overall helpfulness of results	E Q
overall helpfulness of interface	E Q
clicked results	I
ignored results	I
time spent at detailed result view	I
Privacy-Aspects	
sensitivity level of personal information	E Q
disclosure level of personal information	E Q

To collect ground truth data for a *content consumption* task we asked users to annotate web pages with retrieved resources (predefined task "annotate a web page"). Following the assumption that resources are relevant for fragments of web pages only [12], we used annotations to identify the mapping between a selected text fragment of the web page and the most relevant resource for this fragment. Also, because of the selection, we can identify the relation between the text fragment and the corresponding search query, which later can be exploited to automatically generate search queries. To collect data for *content creation*, we asked users to write a blog entry and integrate search results in their text (predefined task "write a blog entry"). In order to collect a greater variety of tasks we encouraged users to engage in regular web browsing behavior in between our tasks of interest, e.g., performing tasks like "watching a funny video clip" or "reading online news". In these tasks users were not required to use the search interface, nor to inspect or rate results. A more detailed description of the tasks is provided in form of a transcript of the instructions for participants along with the dataset[1].

4.2 Procedure and Participants

The data acquisition starts with a general introduction including a demonstration of the interface, followed by explanations about the tasks and privacy issues. Then participants perform the study by executing a combination of predefined tasks (our tasks of interest) and arbitrary web tasks, alternating between those two. For each task participants indicate task specific information, like the task label, their experience level, associated topics. During task execution all interactions are logged. Finally, participants fill out the post-study questionnaire.

8 German speaking students participated in the data collection, 5 male and 3 female. The average age was 27 years, ranging from 21 to 34 years. 2 participants considered themselves as computer experts, 6 as average computer users. 7 participants stated that they were heavy computer and Internet users with a usage of more than 2 hours daily, one participant used the Internet and computer daily, but not more than 2 hours per day.

4.3 Prototype

We implemented the prototype as an extension to Google's Chrome browser, which we describe briefly in this section. We refer the reader to the screencast of a user performing an exemplary task[1] to gain deeper insights. As back-end for search queries, we use the Europeana API[2]. Figure 3 shows a screenshot of the extension, which is injected in every web page, simulating the behavior of a sidebar. The user can switch the injection on or off respectively by clicking the extension icon ($\boxed{01}$). On top of the sidebar the user can define queries ($\boxed{02}$) and select the preferred language of the results ($\boxed{02a}$). The received results are displayed as a list, with a short summary for each result item ($\boxed{03}$). Clicking on a result retrieves the detail view, which is shown as an overlay on the current page. Rating icons are available for each resource ($\boxed{04}$) and the source URL of the resource can be retrieved by the button at $\boxed{05}$. Already existing annotations are highlighted in the text ($\boxed{06}$). For a current text selection, comments ($\boxed{07}$) and the respective resource URI ($\boxed{08a}$) can be added. The URI is automatically copied to the annotation's interface when icon $\boxed{08b}$ is clicked.

The smaller window in figure 3 shows the content of the task tab. Here, the user can select the task type ("annotate a web page", "write a blog entry" or "other") at $\boxed{09}$. For the latter, the user is asked to provide a custom label after task execution, and an additional check box ($\boxed{10}$) is shown to indicate, whether recommendations are desirable for this task. The user can adjust her level of expertise on the task at hand with the slider at $\boxed{11}$ on a scale from 0 (lowest) to 10 (highest). The topics related to the specified task are defined at $\boxed{12}$. This input field features auto-completion for dbpedia-categories, i.e., the user gets suggested dbpedia-categories that possibly match her input while typing. The language for auto-completion can be selected at $\boxed{13}$ (currently supported are French, English and German). After a task has been started ($\boxed{14}$), it is not possible to change its type anymore, while all other settings may still be adjusted, because they might not be known in advance. For instance, the topics related to a task are usually discovered during its execution and the task can be labeled more meaningful.

To support annotation in a content creation task we used the Dokuwiki software[3] and asked users to write simple Wiki syntax and include the retrieved search results manually. Technically, any Web 2.0 user authoring software could be used within this scenario. The log data is stored locally in an indexed database[4]. Annotations conform to the Open Annotation Data Model[5] and are represented in the JSON-LD[6] format. Storing the data locally is advantageous in terms of network traffic and privacy concerns, as it avoids to transmit data for each and every interaction to be logged and users are able to remove interactions they do not want to show up in the log data before providing them to us.

Several adjustments have been made after a pre-test had been performed by one expert and one non-expert. For instance, following the suggestions we included the possibility to delete the last task session (e.g., if users forgot that the extension was active and visited their home-banking site). Also, users reported not to know in advance, which task they are going to perform for the freely chosen tasks, thus, we shifted the task labeling from before starting the task to after its execution.

4.4 Discussion

The proposed setting entails some particularities that need to be discussed. For instance, we treat the search system as a black box and deliver results in the same sequence as we receive them through the respective API. This might bias the ratings and annotations towards higher ranked results. We counterbalance this bias by requesting explicit negative ratings for items and encouraging users to look further through the result list in order to find interesting results.

Because we want to collect training data from users for a systems that does not yet exist, the setting does not reflect completely normal web behavior. In particular, we ask users to formulate queries and annotate page contents with the retrieved results, in order to automatize retrieval of relevant resources in the future, based on the collected data. Also, the task selection is constrained in two ways. First, users avoided tasks in which they had privacy concerns (e.g., doing home banking, visiting specific web sites). Second, the alternation between predefined tasks (website annotation or creating a blog entry) and web tasks of users' choice is artificial. To alleviate the above mentioned problems, we kept the assignment description as near to normal web behavior as possible while still satisfying the constraints imposed by collection of necessary data. Also the prototypical implementation aims to support collection of high quality data with interrupting the normal work flow as little as possible (e.g. no changes in browser tabs for visiting details of recommended items).

Although the number of test users is small, we think that our data set has its benefit by its feature-richness and controlled data collection methodology, empowering application scenarios that require a deep insight into user behavior. An example of such an application, exploiting the feature-richness of the dataset, is presented in section 6. In addition, the insights gained from the dataset can be used for developing a first usable prototype with which additional (less

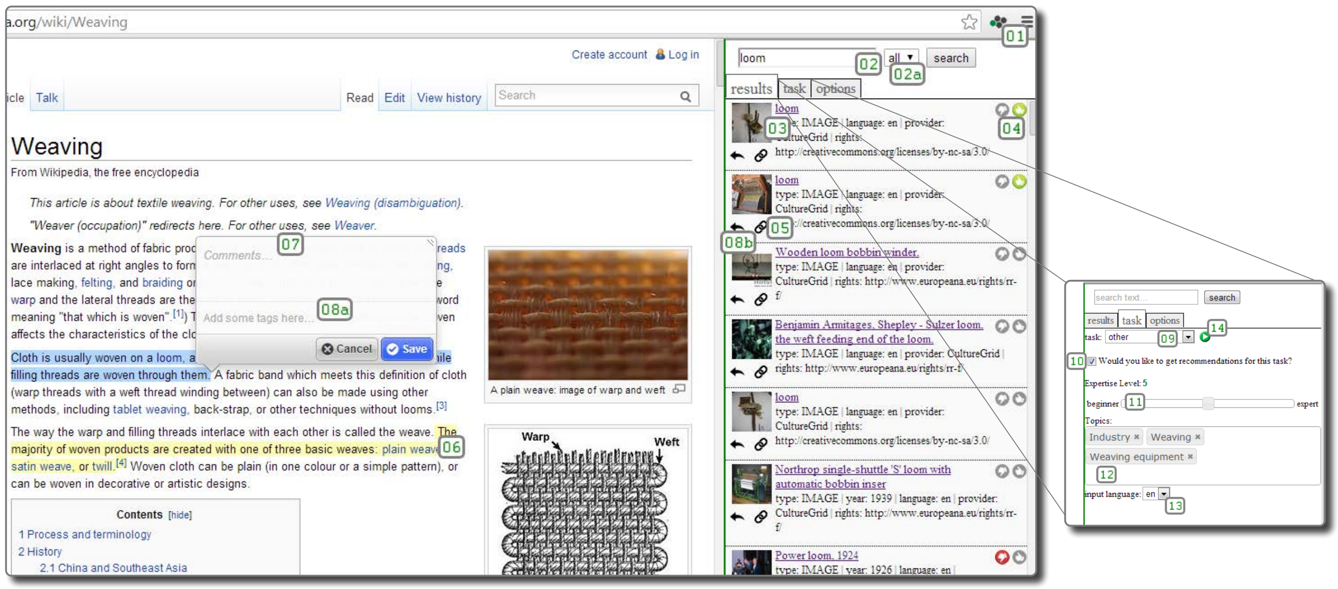
[2]`http://pro.europeana.eu/api`

[3]`https://www.dokuwiki.org/`

[4]`http://www.w3.org/TR/IndexedDB/`

[5]`http://www.openannotation.org/spec/core/`

[6]`http://json-ld.org/`



a.org/wiki/Weaving

Create account Log in

icle Talk Read Edit View history Search

Weaving

From Wikipedia, the free encyclopedia

This article is about textile weaving. For other uses, see Weaving (disambiguation).

"Weaver (occupation)" redirects here. For other uses, see Weaver.

Weaving is a method of fabric proc reads are interlaced at right angles to forn ng, lace making, felting, and braiding or e warp and the lateral threads are the word meaning "that which is woven".[1] T ven affects the characteristics of the clo

Comments...

Add some tags here...

Cancel Save

Cloth is usually woven on a loom, a hile filling threads are woven through them. A fabric band which meets this definition of cloth (warp threads with a weft thread winding between) can also be made using other methods, including tablet weaving, back-strap, or other techniques without looms.[3]

The way the warp and filling threads interlace with each other is called the weave. The majority of woven products are created with one of three basic weaves: plain weave satin weave, or twill.[4] Woven cloth can be plain (in one colour or a simple pattern), or can be woven in decorative or artistic designs.

Contents [hide]

1 Process and terminology

2 History

2.1 China and Southeast Asia

A plain weave: image of warp and weft

loom all search

results task options

loom
type: IMAGE | language: en | provider: CultureGrid | rights: http://creativecommons.org/licenses/by-nc-sa/3.0/

loom
type: IMAGE | language: en | provider: CultureGrid | rights: http://creativecommons.org/licenses/by-nc-sa/3.0/

Wooden loom bobbin winder.
type: IMAGE | language: en | provider: CultureGrid | rights: http://www.europeana.eu/rights/rr-f/

Benjamin Armitages, Shepley - Sulzer loom, the weft feeding end of the loom.
type: IMAGE | language: en | provider: CultureGrid | rights: http://www.europeana.eu/rights/rr-f/

loom
type: IMAGE | language: en | provider: CultureGrid | rights: http://creativecommons.org/licenses/by-nc-sa/3.0/

Northrop single-shuttle 'S' loom with automatic bobbin inser
type: IMAGE | year: 1939 | language: en | provider: CultureGrid | rights: http://www.europeana.eu/rights/rr-f/

Power loom, 1924
type: IMAGE | year: 1926 | language: en |

search text... search

results task options

task: other

[✓] Would you like to get recommendations for this task?

Expertise Level: 5

beginner expert

Topics:

Industry × Weaving × Weaving equipment ×

input language: en

Figure 3: Screenshot of the test data acquisition user interface. Normal browser window content (left, Wikipedia page on weaving), browser extension (right pane), available settings of the option panel (right, center)

Table 2: Data set overview. Time in minutes.

users	tasks	anno	queries	views	ratings	time
8	217	1332	4562	3267	15043	10252

feature-rich data due to privacy concerns) can be collected from a large number of users.

5. DATA SET ANALYSIS

In this section we provide a qualitative and quantitative analysis of the data set we collected. First, we provide a general overview and then we provide detailed analyses for predefined tasks (section 5.1), custom tasks (section 5.2) and the relation between the selected text and the query issued for the selection (section 5.3).

Table 2 provides an overall summary of the collected data. Column "views" refers to the number of times users opened the detail page of results. We count each single view and do not distinguish between different results or one result that has been opened multiple times. Column "time" accumulates the duration (in minutes) of all performed tasks, "anno" counts the total number of users' annotations of web pages with any of the retrieved results. Interestingly, the amount of ratings is more than three times larger than the amount of result views. It is obvious that a lot of results have been rated solely by the short summary as provided in the result list. A potential reason for this is that users assessed the quality of additional information provided in the detailed view as below average in the questionnaire (∅2.75 on a 1-5 scale).

5.1 Predefined Tasks

This section provides an analysis of the collected data from a task-centric perspective, aggregated over all users.

Table 3 presents statistics for the predefined tasks. Tasks prefixed with "A" are content consumption tasks and tasks prefixed with "B" refer to content creation tasks. For each task, annotations were made on one single web page, either within a web site or within one Wiki page. The total number of annotations varies considerably for each task, ranging from 55 (T.B2, wiki entry about an important person) to 112 (T.A8, English Wikipedia article about the Berlin Wall). However, the annotations per user and task do not vary significantly (mean = 11.5, SD = 1.3).

The average task duration for predefined tasks was about 74 minutes per user, with a minimum of 46 minutes (task A5.de) and a maximum of 121 minutes (task B1). Users needed about twice as much time for content creation tasks (112 minutes on average) than for content consumption tasks (65 minutes on average). This tendency is also reflected in the number of clicks amounting to 63 per task per user for content consumption and 224 for content creation tasks.

We collected 8,091 positive and 6,826 negative ratings, amounting to 14,917 ratings. This amount is explained by the assignment asking participants to rate at least 10 results for each search. On average, users rated equally positive and negative, we found no significant difference between the number of positive and negative ratings (Shapiro-Wilks test for normality, $W = 0.9305, p = 0.2777$ for negative, $W = 0.9608, p = 0.7063$ for positive ratings, paired T-Test at confidence level $\alpha = 0.05$: $t = 1.5687, df = 14, p = 0.139$). The column "pages visited" lists the number of visits beyond the predetermined page (i.e., the page to annotate or the page on which a wiki entry had to be created) and result page visits.

5.2 Custom Tasks

Apart from altogether 114 different predefined task executions, users performed 103 freely chosen tasks, and indicated that for 76 of them they would have liked recommendations. After grouping obviously equal tasks, a set of 18 distinct tasks remained. For example, "watch online video", "watch-

Table 3: Task-centric statistics for predefined tasks.

task id	webpage topic	language	annotations	ratings -	ratings +	time [min]	clicks	queries issued	results viewed	pages visited	expertise [0-10]	users
T.A1	wikipedia: Munich	de	98	554	645	586	412	347	387	8	3.4	8
T.A1	wikipedia: Munich	en	81	487	481	430	381	315	230	11	3.2	8
T.A2	wikipedia: World War I	de	97	625	414	467	354	311	165	32	3.4	8
T.A2	wikipedia: World War I	en	96	588	463	461	373	219	147	12	4.0	8
T.A3	warfarehistorian.blogspot.de: warefare	en	108	468	846	705	991	596	294	25	2.1	8
T.A4	wikipedia: Women in Workforce	en	92	557	606	586	559	373	176	54	2.6	8
T.A5	habsburger.net: Franz III & Napoleon	de	86	681	295	371	519	271	217	4	3.5	8
T.A6	wikipedia: Pope	de	91	662	448	467	497	356	229	18	3.4	8
T.A6	wikipedia: Pope	en	92	552	687	601	587	302	298	14	2.9	8
T.A7	britannica.com: French Revolution	en	92	588	554	461	495	238	228	35	4.0	8
T.A8	wikipedia: Berlin Wall	de	99	761	290	483	502	283	322	11	6.2	8
T.A8	wikipedia: Berlin Wall	en	112	702	524	610	423	291	261	12	5.9	8
T.B1	sights of a city	de	60	325	180	729	1612	144	97	142	6.0	6
T.B2	important person	de	55	294	211	692	1322	187	108	73	6.4	6
T.B3	historic event	de	63	247	182	609	1101	279	90	199	5.4	6
total			1322	6826	8091	8266	10128	4512	3249	650	∅ 4.2	8 (max)

Some annotations and ratings were given in tasks, which users performed by their own choice. Those are not include in this table, and therefore the totals differ from the values in the overview table of the previous section.

ing a youtube video" and "watch video on YouTube" were grouped together into "watch online video", whereas "watch news online" and "reading an article" were kept as distinct labels.

The distribution of durations varies considerably across tasks, with a mean of 19, minimum of 3, maximum of 90 and standard deviation of 17 minutes. The strong variation can also be observed within the same task, e.g. the shortest duration of the task "reading online news" (which was executed 43 times) is 3 and the longest duration 47 minutes. This indicates, that task detection is a challenging problem, which cannot be solely solved by a time frame approach. Moreover, we found a disagreement between users on whether they would like to have recommendations for a particular task. This disagreement was also observed for a single user's multiple executions of the same task, likely due to different context of execution. For example, when reading online news, a user may be familiar with one topic, while she is not with another. Thus, the decision when to present recommendations needs to account for features beyond the task at hand. Indicators for a user's familiarity with a topic may be the click rate or dwell time on a particular page.

The two most common tasks ("reading online news" and "watch online video") account for 61% of task executions (excluding predefined tasks), suggesting that a small set of reoccurring tasks constitutes the main part of a user's habitually performed tasks. Although this is an interesting finding within the collected dataset, the rather small number of reoccurring tasks may be caused by the data collection setup (see discussion in section 4.4).

5.3 Selections and Queries

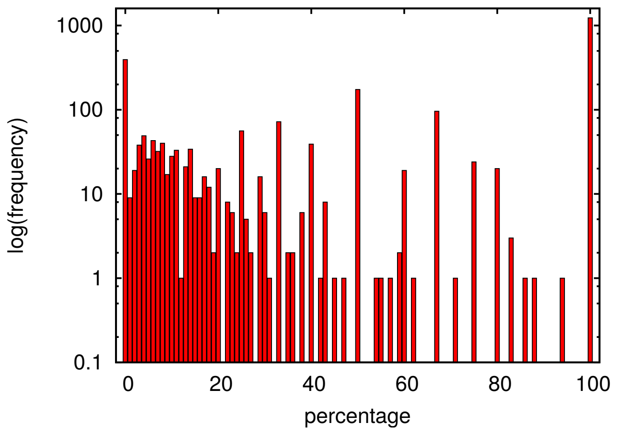
We analyzed the distribution of query terms and selected texts for the content consumption tasks (there was no text selection in the content creation setting). Generally, users preferred to issue queries containing only few terms. The majority of the queries (81%) contains one to three terms with an average query length of 2.8 terms. These numbers are similar to findings from general search engine usage [26]. The text selections were larger, with an average of 8.3 terms. The majority of the selections contains 1 to 4 terms (60%). 10% of the selections had more than 25 terms. 24 ($\approx 1\%$) selections had more than 60 terms.

Figure 4 shows the relation of query terms and terms in the selected text. Figure 4a shows the percentage of terms in the selection contained in the query, i.e., a selection consisting of only query terms counts for the 100% bar, a selection where half of the terms consist of query terms counts for the 50% bar. The majority of selections (46%) consists of query terms only , but 15% of selections contained no query term at all. Figure 4b shows the percentage of query terms contained in the selection, i.e, a selection containing all query terms counts for the 100% bar, a selection where half of the query terms are contained counts for the 50% bar. The majority of selections contained all query terms (57%), but 15% of selections contained no query term at all.

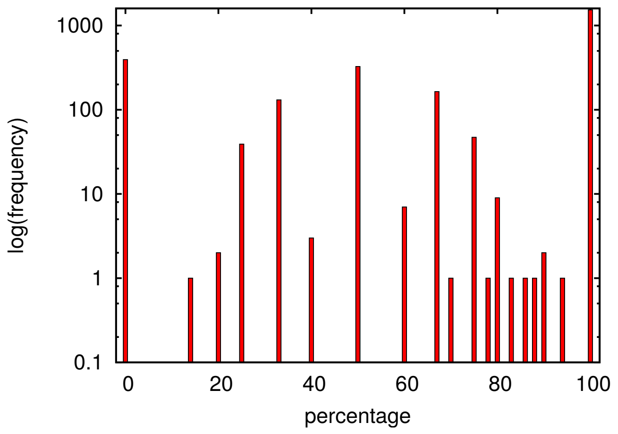
6. LEARNING QUERIES

The findings of the previous section indicate that in 15% of the cases there is no syntactic overlap between the selected text and the query terms and thus, for this cases it is impossible to derive any information for the query directly from the text. Then again, in the majority of cases, information about which terms to use for a query is already contained in the selected text. Therefore, we try to infer, which terms of the selection should actually be used for a query by labeling them as either relevant or not relevant. To predict the relevant terms, we applied a linear-chain conditional random field model (CRF) [10]. This model predicts a sequence of labels for a given input sequence, conditioned on the input features, i.e., it models the probability of the output variables (the labels), conditioned on the observed variables (the input features). We opted for a CRF, as it is advantageous over a Hidden Markov Model in efficiently accounting for dependencies among input features.

For the evaluation, we considered only those text selections, in which at least one term was contained in the corresponding query, amounting to 2449 sequence-query pairs. The input features used were the identity of the term (i), i.e., the term itself, whether the term starts with upper- or lowercase (c), an indicator if the term is a stop word (s), the part-of-speech (POS) tag (t), and the equivalent input features of the preceding (s_{-1}, c_{-1}, t_{-1}) and succeeding (s_{+1}, c_{+1}, t_{+1}) term. A single sample in the training set therefore consists



(a) Ratio of selection terms in query



(b) Ratio of query terms in selection

Figure 4: Term analysis for queries and selected text

of a text selection enriched with the aforementioned features and a label for each term of the selection, which indicates if the term is also contained in the corresponding query (and hence considered relevant).

The list of stop words is the one provided by the "tm" package for R[7], the POS tags were obtained with NLTK [3] and the CRF models were computed with Mallet [16]. We evaluated the performance of 29 feature combinations using 10-fold cross-validation. In order to evaluate the stability across users and tasks we also performed cross-validation on splits defined by users (all but one user as training and one user for test), and tasks respectively.

Table 4: Accuracies [%] for query prediction from selected text. Cross-validated using splits over users, tasks, and 10-fold random.

		feature set			trivial	
		i, c, t	i, t	c, t	rejector	acceptor
users	mean	76	77	75	51	49
	SD	15	15	18	35	35
tasks	mean	82	83	82	71	29
	SD	6	6	7	8	8
10-fold	mean	89	88	84	71	29
	SD	1	2	1	2	2

i - the identity of a term, i.e. the term itself
c - whether the term begins with upper- or lowercase
t - POS tag

The best performing feature combinations are shown in table 4. As the CRF model assigns a label to each term in the selection (identifying it as relevant or not relevant), accuracy refers to the ratio of correctly labeled terms to the total number of terms. Incorporating a term itself as a feature (i, c, t & i, t) leads to the best results, but this may not generalize well due to the limited vocabulary in the dataset. Nevertheless, feature combinations without the words provide similar results as well (e.g., the combination of case-identifier and POS-tag, c, t) and thus are the better option.

The standard deviations reveal, that the query behavior is stable over tasks, but not over users. In fact half of the users incorporated the major part of the selection into their queries and the queries of the other half contained only a minority of the selection terms. Thus, prediction performance drops for the evaluation over users.

[7]`http://cran.r-project.org/web/packages/tm/`

7. RELATED WORK

Providing long-tail recommendations is a highly challenging task, first of all because of the data sparsity issue: only a few or even no ratings are available for items in the long-tail. To overcome this problem, the authors in [18] partition the whole item set into head and tail parts and cluster the items in the tail. In [11] recommendations are obtained by combining the items in a user's personal long-tail with users, which have those items in their head portion. While these approaches still require the existence of at least a few ratings in the tail or even the existence of dense data in the head, Stickroth et al. [25] aim to provide high quality recommendations in a network with a small amount of users and items (and hence without the presence of a dense head). Therefore the authors propose a multilevel approach, with a decreasing degree of personalization and different recommendation strategies at each level. Their dataset encompasses 60 ratings on 151 items by 175 users and is not published. Closest to our work, Wang et al. [27] conducted a user study in the cultural heritage domain in which they elicited user models with ratings of museum objects of the Rijksmuseum Amsterdam from 39 participants.

Most of the approaches for user data collection for long-tail domains use server-side data logging. A representative example is the smartmuseum approach were user interests are either manually given or by tagging and rating of resources [23]. A game-based approach to server side collection was pursued by Wang et al. [27] who used an interactive quiz to collect ratings for museum objects. Goecks and Shavlik [6] use client-side data collection in a Web browser for user interest detection based on the text of the webpage, clicked hyperlinks, scrolling and mouse activity.

All of those data sets capture the features we identified as necessary to collect only partly. To the best of our knowledge, there is no publicly available dataset, which accounts for the specific challenges of long-tail recommendations and contains the required data.

8. SUMMARY AND FUTURE WORK

In this paper, we analyzed feature requirements for search personalization with respect to long-tail domains. Based on this analysis, we provide a prototype for collecting the required data from web users. Further, we collected a data set with this prototype, which comprises over 170 hours of user interaction and is publicly available. We provided an in-depth analysis of this data set. With the automatic generation of queries from text selections, we presented an exemplary application of the data set. The data collection is easily adaptable to other domains by modifying the call to the search system in the prototype (currently Europeana).

We intent to provide a modified browser extension to the general public. Depending on the individual privacy settings, we will collect more, but less feature-rich test data and further try to infer user model information from this data (e.g. task and topic detection). Given the promising results for query generation, we will investigate this area in more detail and try to enhance the query prediction by incorporating additional context beyond the selection, e.g., from the browsing history, and improve the result set by query modification.

9. ACKNOWLEDGMENTS

The presented work was developed within the EEXCESS project funded by the European Union Seventh Framework Programme FP7/2007-2013 under grant agreement number 600601.

10. REFERENCES

[1] G. Adomavicius and A. Tuzhilin. Context-aware recommender systems. In *Recommender systems handbook*, pages 217–253. Springer, 2011.

[2] A.-L. Barabási, R. Albert, and H. Jeong. Scale-free characteristics of random networks: the topology of the world-wide web. *Physica A: Statistical Mechanics and its Applications*, 281(1–4):69 – 77, 2000.

[3] S. Bird, E. Klein, and E. Loper. *Natural language processing with Python*. O'Reilly Media, Inc., 2009.

[4] Z. Dou, R. Song, and J.-R. Wen. A large-scale evaluation and analysis of personalized search strategies. In *Proc. WWW*, 2007.

[5] N. Fuhr. Probabilistic models in information retrieval. *The Computer Journal*, 35(3):243–255, 1992.

[6] J. Goecks and J. Shavlik. Learning users' interests by unobtrusively observing their normal behavior. In *Proc. IUI*, 2000.

[7] M. Granitzer, C. Seifert, S. Russegger, and K. Tochtermann. Unfolding cultural, educational and scientific long-tail content in the web. In *UMAP Extended Proceedings*, 2013.

[8] D. Hauger, A. Paramythis, and S. Weibelzahl. Using browser interaction data to determine page reading behavior. In *Proc. UMAP*, 2011.

[9] G. Koutrika and Y. Ioannidis. Rule-based query personalization in digital libraries. *International Journal on Digital Libraries*, 4(1):60–63, 2004.

[10] J. D. Lafferty, A. McCallum, and F. C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. ICML*, 2001.

[11] K. Lee, W. S. Yeo, and K. Lee. Music recommendation in the personal long tail. In *Proc. WOMRAD, ACM RecSys*, 2010.

[12] X. Li, T.-H. Phang, M. Hu, and B. Liu. Using micro information units for internet search. In *Proc. CIKM*, 2002.

[13] Y. Lindell and E. Waisbard. Private web search with malicious adversaries. In *Proc. Int. Conf. on Privacy Enhancing Technologies*, 2010.

[14] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008.

[15] G. Marchionini and R. White. Find what you need, understand what you find. *Int. J. Hum. Comput. Interaction*, 23(3):205–237, 2007.

[16] A. K. McCallum. Mallet: A machine learning for language toolkit. http://mallet.cs.umass.edu, 2002.

[17] A. Micarelli, F. Gasparetti, F. Sciarrone, and S. Gauch. Personalized search on the world wide web. In P. Brusilovsky, A. Kobsa, and W. Nejdl, editors, *The adaptive web*, chapter Personalized search on the world wide web, pages 195–230. Springer-Verlag, Berlin, Heidelberg, 2007.

[18] Y.-J. Park and A. Tuzhilin. The long tail of recommender systems and how to leverage it. In *Proc. of the RecSys '08*, 2008.

[19] J. Pitkow, H. Schütze, T. Cass, R. Cooley, D. Turnbull, A. Edmonds, E. Adar, and T. Breuel. Personalized search. *Commun. ACM*, 45(9):50–55, Sept. 2002.

[20] N. Ramakrishnan, B. J. Keller, B. J. Mirza, A. Y. Grama, and G. Karypis. Privacy risks in recommender systems. *IEEE Internet Computing*, 5(6):54–62, Nov. 2001.

[21] B. J. Rhodes. *Just-In-Time Information Retrieval*. PhD thesis, Massachusetts Institute of Technology, 2000.

[22] J. J. Rocchio. Relevance feedback in information retrieval. In G. Salton, editor, *The Smart retrieval system - experiments in automatic document processing*, pages 313–323. Englewood Cliffs, NJ: Prentice-Hall, 1971.

[23] T. Ruotsalo, E. Mäkelä, T. Kauppinen, E. Hyvönen, K. Haav, V. Rantala, M. Frosterus, N. Dokoohaki, and M. Matskin. Smartmuseum: Personalized context-aware access to digital cultural heritage. In *Proc. ICSD*, 2009.

[24] M. Shokouhi and L. Si. Federated search. *Found. Trends Inf. Retr.*, 5(1):1–102, 2011.

[25] S. Strickroth and N. Pinkwart. High quality recommendations for small communities: the case of a regional parent network. In *Proc. Recsys*, 2012.

[26] M. Taghavi, A. Patel, N. Schmidt, C. Wills, and Y. Tew. An analysis of web proxy logs with query distribution pattern approach for search engines. *Comput. Stand. Interfaces*, 34(1):162–170, Jan. 2012.

[27] Y. Wang, L. M. Aroyo, N. Stash, and L. Rutledge. Interactive User Modeling for Personalized Access to Museum Collections: The Rijksmuseum Case Study. In *User Modeling*. Springer Berlin Heidelberg, 2007.