Memory-based Recommendations of Entities for Web Search Users

Ignacio Fernández-Tobías
Universidad Autónoma de Madrid
Madrid, Spain
ignacio.fernandezt@uam.es

Roi Blanco
University of A Coruña
A Coruña, Spain
rblanco@udc.es

ABSTRACT

Modern search engines have evolved from mere document retrieval systems to platforms that assist the users in discovering new information. In this context, entity recommendation systems exploit query log data to proactively provide the users with suggestions of entities (people, movies, places, etc.) from knowledge bases that are relevant for their current information need. Previous works consider the problem of ranking facts and entities related to the user's current query, or focus on specific recommendation domains requiring supervised selection and extraction of features from knowledge bases. In this paper we propose a set of domain-agnostic methods based on nearest neighbors collaborative filtering that exploit query log data to generate entity suggestions, taking into account the user's full search session. Our experimental results on a large dataset from a commercial search engine show that the proposed methods are able to compute relevant entity recommendations outperforming a number of baselines. Finally, we perform an analysis on a cross-domain scenario using different entity types, and conclude that even if knowing the right target domain is important for providing effective recommendations, some inter-domain user interactions are helpful for the task at hand.

Keywords

Recommender systems, Web Search, Entity Recommendation

1. INTRODUCTION

Search engines have been evolving into sophisticated tools that assist their users in satisfying a rapidly increasing variety of information needs. For that purpose, modern search systems mine session trails in query logs and extract patterns of interactions that allow a better understanding of the users' interests [12]. This has enabled the development of new techniques that enhance the overall search experience, such as query recommendation and search result personalization [24, 25, 26, 28]. Regardless of its potential, the nature of the data typically available to a search engine presents a number of challenges for personalization algorithms. Recommendation techniques such as Collaborative Filtering [11], widely used in e-commerce systems, are difficult to apply in web search due to the huge size of the document collection and the extreme data sparsity.

CIKM'16 , October 24 - 28, 2016, Indianapolis, IN, USA
© 2016 Copyright held by the owner/author(s). Publication rights licensed to ACM. ISBN 978-1-4503-4073-1/16/10...$15.00
DOI: http://dx.doi.org/10.1145/2983323.2983823

Nonetheless, there have been significant advances in understanding and helping users with their information needs. In addition to matching query terms, commercial systems are now able to detect the intention or semantics behind search queries, and directly provide answers in the results page [4, 5]. These systems make use of knowledge bases (also called knowledge graphs) that contain structured data with facts about millions of real-world entities, together with their attributes and relations. When a query is submitted, the search engine extracts named concepts from the query string and links them to entities in the knowledge base. In addition to the traditional links, the system returns information about the entities named in the query, together with suggestions about other entities related in the knowledge base. The goal is to promote exploratory search and assist the users in situations where their information need is not clearly defined [19].

Previous work has studied the problem of extracting named entities from the queries and linking them to knowledge bases [6, 10, 20], as well as ranking the related entity suggestions [5]. These approaches have been shown to increase the users' satisfaction and engagement with the system [19]. However, their analysis is limited to the current query, and typically disregard the users previous interactions with the system, thus only providing non-personalized suggestions. More recently, some approaches have considered the full user history to personalize the ranking of recommended entities related to the query.

Rather than suggesting entities related to the current query, in this work we aim to further assist the user in her exploration task, more generally providing personalized entity recommendations tailored to her current search session. Our goal is to help the user discover new information by delivering suggestions of relevant entities, without requiring her to launch additional search queries. Consider for example a user that is interested in learning about Artificial Intelligence. The user submits the query *artificial intelligence* and clicks on the corresponding Wikipedia article. An entity recommendation system could then suggest related entities such `Natural_language_processing` and `Machine_Learning`. Consider now another user that submits the same query *artificial inteligence* and clicks on the entity `A.I._Artificial_Intelligence` (the 2001 science fiction film). The system would detect that the user is interested in movies and instead recommend another sci-fi movie `Minority_Report_(film)`. An additional benefit of a system capable of delivering such personalized recommendations would be to provide suggestions of entities that could take a large effort from the user to find in the huge knowledge bases.

In this paper we propose a number of memory-based methods inspired by nearest neighbor techniques [11] that exploit the users' sessions in a collaborative fashion for computing personalized entity recommendations. We benchmark three families of novel techniques that respectively make use of documents, queries, and sessions using a large query log extracted from a commercial search engine, and discuss which properties characterize the most successful ones. One advantage of our memory-based approaches as opposed to model-based methods is that they are able to scale to larger amounts of training data and handle new users easily.

In contrast to previous work that proposed solutions that require specific domain knowledge [31], our methods are general and suitable for any type of entity. Moreover, we are interested in understanding how the quality of recommendations depends on the domain of the entities in user sessions in a cross-domain setting. In particular, we investigate whether it is beneficial to exploit other types of entities when recommending, e.g. movies to a user or if, on the other hand, it is better to only consider her past preferences about movies.

To the best of our knowledge this is the first work that performs entity recommendations at Web scale using a purely collaborative filtering approach suitable for any type of entities. We also highlight the fact that even if our experimental evaluation is based on recommending entities from a closed-world reference knowledge base (Wikipedia), the algorithms and techniques described in the paper could generally apply to personalized within-sessions document click prediction.

The rest of the paper is organized as follows. Section 2 discusses related work, Section 3 introduces the problem of personalized entity suggestion, our particular formalization, and a number of recommender-system inspired methods to solve the problem. Sections 4 reports on experimental results, and in Section 5 we study the performance of our methods in cross-domain recommendation scenarios. The paper concludes in Section 6.

2. RELATED WORK

In this paper we are interested in the task of providing personalized recommendations of entities to users of a search engine. Therefore, we briefly review in this section related work on (i) semantic search and entity ranking, (ii) personalization in search, and (iii) recommender systems.

Entity and semantic search. Semantic search deals with the problem of finding relevant information in knowledge bases given an unstructured keyword-based query. The information in the knowledge base comprises different useful acts about the so-called *entities* (people, locations, organizations, or products), which can be ingested from data sources such as Wikipedia or Freebase, whereas others are extracted from unstructured text. An important task in semantic search is to retrieve resources in a knowledge base referred by named entities in the query [20].

Entity search is viewed as an ideal paradigm to support exploratory search. It provides semantically rich answers – entities and their relations, which are often considered more suitable for search exploration than individual web pages. In this work we will be dealing with entity to entity and entity to query similarity functions, which have been extensively studied previously. The problems of measuring entity similarity, and retrieving entities related to an input entity or query, have been tackled in several works [9, 10, 13, 14] by building graphs of entities and their relations, and applying different types of computations on these graphs. Other approaches [1, 30] build *entity-relationship* models, where entities take part (with various roles) in different types of relations representing real-world associations.

Blanco et al. [5] present a model that ranks entities in a knowledge base to provide suggestions of entities related to the user's query, which is shown to promote exploratory search and user engagement. Miliaraki et al. [19] analyze features of users, queries, and sessions that show an explorative behavior, and train a logistic regression model to predict whether the user will click or not a given recommended entity. Finally, Bi et al. [4] propose a probabilistic three-way model to personalize the ranking of suggested entities. Rather than ranking entities related to the given query, in this paper we are interested in methods that proactively recommend relevant entities for a user's search session in a personalized fashion.

Personalized search. Personalized retrieval systems aim to improve the ranking of search results for each user by exploiting information about her previous interactions or current context [26]. In general terms, tracking user search behavior over time (historical data) has been shown to lead to better prediction of future search behavior e.g. [8, 21, 23].

Sontag et al. [24] develop a probabilistic model that takes into account the user's long-term interests in order to personalize the ranking of search results. Xiang et al. [29] propose a learning to rank model that exploits the short-term context of queries to improve ranking quality. However, the performance of these systems may be degraded when the user's search history is not large enough. In order to address this scenario, White et al. [28] propose a task-oriented model that exploits the behavior of other users that were involved in similar tasks. In addition to improving result ranking, some works addressed the task of predicting future user search interests. White et al. [27] present a study comparing the suitability of different sources of query context to predict the ODP categories of the documents that the user will click in the result page. Our work is similar in that we also exploit the query context for prediction, but we focus on recommending particular items on the much larger set of Wikipedia entities rather than limited topical categories.

Recommender systems. The goal of recommender systems is to assist the users deal with the information overload by proactively suggesting items that are useful to them. Content-based methods [17] extract features from the items and recommend those that are most similar to the ones in the user's history. In contrast, Collaborative Filtering (CF) approaches exploit interactions of like-minded users to compute personalized recommendations. k-nearest neighbors (kNN) [11] methods are among the most popular memory-based approaches to CF, and have been successfully applied in major e-commerce systems [16]. In the context of Web search, Sugiyama et al. [25] adapted kNN to compute user-term weight profiles for personalized retrieval.

Closest to our work is the paper by Yu et al. [31], where the authors described a movie recommendation system for Web search users. Specifically, they addressed the problem of predicting the last entity of a user click log sequence given the previous clicked entities, which possibly include entities clicked in the same query. One important difference with respect to our work is that we take queries into account and always predict which entities will be clicked on the next (different) query, thus anticipating the user's needs. We argue that our task is harder and closer to a real-world scenario, as the user may switch to a different topic in the next query. Furthermore, the method proposed in [31] uses the entities' content attributes to extract features for learning the prediction model. Our methods do not take the content into account and only use the search logs, which makes them suitable for recommending any type of entities as we do not have to extract a different set of features for every entity type. Moreover, in our experiments we are not focused on any particular domain and consider a much larger set of candidate entities for recommendation.

3. MEMORY-BASED ENTITY RECOMMENDATIONS

In the context of exploratory search in the Web, users click and browse entities of many different types from multiple domains. An entity recommender system should be able to address this heterogeneity and deliver suggestions independently of the target domain. Model-based approaches such as that proposed in [31] are able to compute relevant personalized recommendations only once the target domain is known, as they require supervised selection and extraction of good domain-dependent entity features in order to be successful. Aiming to develop a domain-agnostic entity recommender, in this paper we avoid crafting domain dependent features by exploiting the user interactions available in search logs in a collaborative filtering fashion. Rather than comparing the content features of the entities, we leverage patterns of shared users, queries, and sessions to estimate entity-entity similarities. In this section, we present three memory-based approaches inspired by the nearest neighbors methods in the Recommender Systems literature [11], as they provide a flexible framework for collaborative filtering with different types of user interactions (clicked documents, submitted queries, etc.) and scale to large amounts of data.

3.1 Problem formalization

We now provide a probabilistic formalization of the problem of personalized entity recommendations and describe a number of solutions for tackling it. Note that in the following we might use documents and entities interchangeably. We define three families of approaches which operate by finding similar items and then estimating the likelihood of a clicked document in the given session. These techniques are named after the main variable from which the similarity is calculated, namely document, query and session-based. In the following, we assume sessions $s \in \mathcal{S}$ are multi-sets of entities $e \in \mathcal{E}$ and queries $q \in \mathcal{Q}$, and we only require to be able to count occurrences of e and q within s. For the sake of clarity, throughout this section we use the notation $d \in \mathcal{E}$ to refer to documents (entities) clicked in a session ($d \in E(s)$), and we reserve the variable e for candidate entities for recommendation. Our techniques select the most likely relevant entity for a given session once the aforementioned likelihood is estimated:

$$e^* = \operatorname*{argmax}_{e \in \mathcal{E}} p(e|s) \ , \qquad (1)$$

or alternatively, we can generate a ranking of all e sorting by $p(e|s)$. In the following, we will define several alternatives for computing item-to-item similarities and how to aggregate them in a probabilistic fashion in order to recommend an entity e^*.

3.2 Document-based

The intuition behind our first approach is that an entity e is likely to be relevant to the target session s if it is similar to the entities previously clicked by the user within the same session. We estimate the probability of an entity e being relevant for session s as:

$$p(e|s) = \sum_{d \in E(s)} p(e|d)p(d|s) \qquad (2)$$

The factor $p(e|d)$ captures the similarity between pairs of entities, whereas $p(d|s)$ estimates how relevant the entity d is for the task within session s. We compute the pairwise entity similarities $p(e|d)$ in a collaborative fashion, exploiting users interactions in the query logs rather than content information. The main motivation for not using the content is to overcome the heterogeneity of attributes among different types of entities, as we do not want to restrict our analysis to any particular domain. Thus, we consider similarity measures based on co-occurrence of entities within user profiles, queries, and sessions using Jaccard's coefficient.

In order to estimate how relevant a clicked document (entity) d is in session s we take into account all the queries in the session from which it is reached:

$$p(d|s) = \sum_q p(d, q|s) = \sum_q p(d|q, s)p(q|s) \qquad (3)$$

That is, the likelihood of entity d in session s is a convex combination of the importance of result d for each query, $p(d|q, s)$, weighted by the query likelihood of q in the session s, $p(q|s)$. We note that modern search engines are able to adapt the list of retrieved documents for a given query depending on a number of factors, such as the user's previous interactions and context. More importantly, for the purpose of entity recommendation, the notion of relevance we model is specific for each user and session. Hence, we do not drop the conditional dependence on s, and consider three possibilities for $p(d|q, s)$:

- Click-based. All clicked results for the query are equally relevant, $p(d|q, s) = |\{d' | d' \text{ clicked in } q \in \mathcal{Q}(s)\}|^{-1}$ if d was clicked for query $q \in \mathcal{Q}(s)$, and 0 otherwise.
- Dwell time. A document d is likely to be relevant the more time the user spends examining it. We assume that the probability is directly proportional to the dwell time, $p(d|q, s) \propto \mathrm{dt}(d, q, s)$, and normalize using the total dwell time of all the clicked documents for the query.
- Rank-based. This approach assumes that a document d is more likely to be important for the query q if it was ranked higher by the search engine, i.e. $p(d|q, s) \propto \mathrm{rank}(d, q, s)$. We note that this method delegates the probability estimation to the search engine's ranking

algorithm, which may not take into account the rest of the user session s.

In all cases we define $p(d|q,s) = 0$ if the user did not click on d as a result to query q in the session s. Similarly, $p(q|s)$ captures how important query q is for the task carried in session s. In this work we explore two different alternatives: (i) *uniform*, so that all observed queries are equally important, and (ii) *temporal*, which assumes that queries launched earlier in the session are likely to be less representative of the current user task. Aiming to model the topic drift within a session, we define the latter as $p(q|s) \propto e^{-(t_N - t_q)}$, where t_N is the timestamp of the last query in the session and t_q the timestamp of query q. Here we also consider $p(q|s) = 0$ for any query q not observed in s, $q \notin \mathcal{Q}(s)$.

3.3 Query-based

Rather than directly recommending entities similar to those clicked in the target session, our second approach works in two steps. First, it finds queries from other sessions in the search logs that are potentially relevant to the target session, operating in a similar fashion as a query suggestion system. Next, it retrieves entities clicked from those queries. Specifically, we estimate the query-based relevance of candidate entity e as

$$\begin{aligned} p(e|s) &= \sum_q p(e,q|s) = \sum_q p(e|q,s)p(q|s) \\ &= \sum_q p(e|q)p(q|s) \end{aligned} \tag{4}$$

The factor $p(q|s)$ captures the probability that recommended query q is relevant to session s. Note that in contrast to the previous document-based method the query q is now not observed in s, $q \notin \mathcal{Q}(s)$, and instead is chosen among the queries submitted by other users performing similar tasks. The factor $p(e|q)$ measures how relevant is the entity e for a given query q across the rest of the sessions in the search logs. Thus, in the last equality in Equation 4 we are assuming the conditional independence of candidate entity e and the target session s once a particular query q is chosen.

In order to estimate $p(e|q)$, we average over all the sessions in the search logs that contain the query q:

$$\begin{aligned} p(e|q) &= \sum_{t\in\mathcal{S}} p(e,t|q) = \sum_{t\in\mathcal{S}} p(e|t,q)p(t|q) \\ &= \sum_{t\in\mathcal{S}} p(e|q,t)\frac{p(q|t)p(t)}{p(q)} \\ &\propto \sum_{t\in\mathcal{S}} p(e|q,t)p(q|t)p(t) \end{aligned} \tag{5}$$

Here, $p(e|q,t)$ is either click-based, dwell time, or rank-based, and $p(q|t)$ is *uniform* or *temporal* as previously. Note that $p(q|t)$ is the probability of *choosing* a query q *from* session t, $q \in \mathcal{Q}(t)$, not to be confused with the relevance of the candidate query q recommended for target session s. Thus, $p(q|t) = 0$ if $q \notin \mathcal{Q}(t)$. Finally, $p(t)$ is the prior probability of choosing session t, which we assume uniform for simplicity. Nonetheless, more sophisticated options are possible, such as giving higher priority to sessions that ended successfully, or sessions from expert users [28].

We now turn our attention to the query suggestion step for computing the query relevance probability $p(q|s)$ in Equation 4. We have explored three different alternatives, two of which follow a collaborative filtering-like approach, and one that makes use of an entity linker for queries.

3.3.1 Similar query recommendation

The first alternative we considered works analogous to the document-based approach described in Section 3.2, but considering queries rather than entities. Specifically, we select those queries from the search logs that are most similar to the ones in the target session:

$$p(q|s) = \sum_{q_s\in\mathcal{Q}(s)} p(q|q_s)p(q_s|s) \tag{6}$$

Again, we considered $p(q_s|s)$ to be either *uniform* or *temporal*, whereas $p(q|q_s)$ captures the similarity between queries. Among the several options available to compute query similarity, in this paper we focus on approaches that follow the collaborative filtering paradigm. Hence, we estimate $p(q|q_s)$ using Jaccard's coefficient to measure the co-occurrence of queries within (i) user profiles, or (ii) search sessions.

3.3.2 Likely query recommendation

The intuition behind the second query recommendation alternative is that a query q is more likely to be relevant to the target session s the more documents from s it covers, i.e. the more documents from s that are among q's results. In particular, we estimate

$$p(q|s) = \sum_d p(q|d)p(d|s) \propto \sum_d p(d|q)p(q)p(d|s) \tag{7}$$

where we have used Bayes' rule and assumed uniform prior probability for all entities. As previously, the factor $p(d|s)$ is the probability of choosing e among the entities clicked in session s, which we compute using Eq. 3. Likewise, $p(d|q)$ is computed using Eq. 5. Finally, we estimate the prior relevance probability of the query $p(q)$, as either (i) *uniform*, or (ii) *based on popularity*, to favor queries that are more frequently chosen by the users. In the latter, we define $p(q) \propto \log(1 + |S(q)|)$, where $S(q)$ is the set of sessions that contain query q in the search logs.

3.3.3 Linked query recommendation

The last query recommendation method that we analyze makes use of a query entity linker [6] to extract named entities from queries and relate them in a knowledge base. Given a query q the linker returns an entity e_q together with a probability $p(e_q|q)$ that models the likelihood of the match (see [6] for details). We use the output of the entity linker to find which queries are the most related to the entities clicked in the target session:

$$\begin{aligned} p(q|s) &= \sum_d p(q|d,s)p(d|s) = \sum_d p(q|d)p(d|s) \\ &= \sum_d p(q|e_q)p(e_q|d)p(d|s) \end{aligned} \tag{8}$$

Similarly to previously described methods, we compute $p(d|s)$ using Eq. 3. The factor $p(e_q|d)$ models the similarity between the clicked entity d and the candidate query's linked entity e_q, which we compute as in Eq. 2. Finally, we set $p(q|e_q) \propto p(e_q|q)$ using Bayes' rule and assuming uniform prior probabilities for $p(q)$ and $p(e_q)$.

3.4 Session-based

The last family of approaches works by finding similar sessions to the target session and recommending entities from those sessions despite those query and entity trails having been submitted by different users.

$$p(e|s) = \sum_{t \in \mathcal{S}} p(e|t)p(t|s) \tag{9}$$

The main component of the session-based approach is the pairwise similarity modeled through $p(t|s)$, which we compute by using either (i) *entities* clicked in the session or, (ii) *queries* formulated in the session. For each of these we analyze four different methods for computing session pairwise similarity.

3.4.1 Co-occurrence similarity

In this method, sessions s and t are similar if the entities clicked in each are shared. More specifically, we compute the similarity using the Jaccard's coefficient between sets $E(s)$ and $E(t)$. For queries, we compute the similarity using the sets $\mathcal{Q}(s)$ and $\mathcal{Q}(t)$, respectively.

3.4.2 Centroid

The *centroid* session similarity is motivated by the recent developments on document similarity based on word embeddings [15]. We train two different methods where documents correspond to search sessions, and words within documents are either (i) clicked entities, or (ii) submitted queries. We use word2vec [18] to extract a vector $v_d \in \mathbb{R}^D$ for each entity d (respectively v_q for each query q). We refer the reader to [18] for details about word2vec. Then, we compute the similarity between sessions based on the distance between their centroid vectors:

$$p(t|s) \propto \left(1 + \left\| \frac{1}{|E(s)|} \sum_{d \in E(s)} \vec{v_d} - \frac{1}{|E(t)|} \sum_{d \in E(t)} \vec{v_d} \right\| \right)^{-1} \tag{10}$$

Given the size of our dataset (see Section 4.1), in practice we do not compute the centroid similarity for all possible session pairs. Note that, unlike Jaccard's coefficient, the computation of the centroid similarity involves full vectors of high dimensionality ($D = 100$ in our experiments). Instead, we use *Approximate Nearest Neighbors* techniques to find the most similar sessions to s. In particular, we use *Locality Sensitive Hashing* (LSH) based on random projections [2] to select the sessions with the closest centroid to s.

3.4.3 Word Mover's Distance

The third similarity measure is an adaptation for search sessions of the Word Mover's Distance (WMD) proposed in [15]. Specifically, we define the similarity between sessions s and t using the relaxed version of WMD, where each entity (respectively query) in s is mapped to its most similar entity in t:

$$p(t|s) \propto \sum_{i \in E(s)} \sum_{j \in E(t)} \mathbf{T}_{ij} \cos(\vec{v_i}, \vec{v_j}) \tag{11}$$

where $\mathbf{T}_{ij} = p(i|s)$ if $j = \text{argmax}_j \cos(\vec{v_i}, \vec{v_j})$ or 0 otherwise. $\vec{v_i}$ and $\vec{v_j}$ are the word2vec embeddings for entities (queries) i and j. Again, due to the size of our dataset, we do not compute the cosine similarity for all possible pairs of entities (queries), and instead use LSH to find an approximate set of top ten most similar candidates. In the rest of the paper we refer to this method as WMS, as it is adapted for computing similarity rather than distance.

3.4.4 Translation model

The last similarity measure is based on the IBM-1 translation model [3], which defines the probability of translating a sequence of words between languages. Inspired by [28], where the IBM-1 model was used to compute the semantic similarity between two queries, we adapt the same notion to search sessions. In particular, we think of each session as a sequence of words, where words correspond to either entities or queries, and define

$$p(t|s) = \prod_{d_t \in E(t)} \sum_{d_s \in E(s)} p(d_t|d_s)p(d_s|s) \tag{12}$$

Again, $p(d_s|s)$ is the importance of entity d_s in session s, which we compute using Eq. 3, and $p(d_t|d_s)$ captures the similarity between d_t and d_s. In our experiments we use word embeddings like in previous subsections and compute $p(d_t|d_s) \propto \cos(\vec{d_t}, \vec{d_s})$, as we obtained better results than using Jaccard's coefficient in preliminary tests. The translation model for queries is analogous, replacing $d_t \in E(t)$ and $d_s \in E(s)$ with $q_t \in \mathcal{Q}(t)$ and $q_s \in \mathcal{Q}(s)$, respectively.

4. EXPERIMENTAL WORK

4.1 Dataset

All the experiments reported are carried out on a sample spanning three months (April – June, 2013) of user click data (US market) coming from a large commercial search engine log. Although most of our methods do not impose a restriction in what types of entities are being recommended to users, in the forthcoming experiments we will restrict the set-up to suggest entities from a knowledge base. This has the advantage of keeping the amount of possibilities limited (to the order of millions or tens of millions) as well as having a real-world application in promoting exploratory search behavior [19].

We retained only clicks to articles in the English Wikipedia, which is one of the largest repositories of knowledge freely available on the web. Moreover, the DBpedia project[1] provides a well-structured and machine-readable version of each article (entity) in Wikipedia. We make use of DBpedia to extract the entity type (the `rdf:type` field). However, it is worth to note that our methods rely on collaborative features rather than content, and we only use the content attributes to classify entities in different domains.

Sessions in our dataset are delimited based on a 30 minute time-out. Since we are interested in recommending entities to users based on their interactions within a given session, we discarded from our analysis sessions that contain a single query. These consist mostly of navigational queries, where users are not engaged in an exploration task. We also removed from our dataset entities that received less than 50 clicks which greatly reduced the item set to a manageable size for training, without affecting the coverage too much. After preprocessing, our dataset contains about 7M sessions, over 4M users, 6M queries, 300k entities, and more than 19M clicks. We split the data into one-month folds and used them as follows: the first month for training the algorithms, the

[1]http://dbpedia.org

second month for validation/tuning, and the third month for testing. Regarding the evaluation methodology, we further split each session in the test set into two parts. This split enables us to simulate a real user interacting with the search engine for whom the system has to compute entity recommendations. The first chunk of the session, which is used as input to the recommendation algorithms, contains the clicked documents from all but the last query. The second part contains clicks only from the last query, and is used as the ground truth to estimate the performance of the system.

4.2 Baselines and metrics

We compared the performance of our proposed approaches for entity recommendation against the following algorithms:

- **Most popular**. Non-personalized method that recommends the most clicked entities, provided the user has not already seen them in the target session.
- **Daily popular**. Recommends the entities that received the most clicks on the same day the target session was started.
- **Next entity**. Suggests the entities that are most frequently clicked immediately after the last browsed entity in the target session.
- **Item kNN**. Provides personalized recommendations of entities similar to those in the user's full previous search history [22]. Entity similarity is computed using Jaccard's coefficient over the sets of users that clicked the entities.
- **PRM-KNN**. Personalized entity recommendation model proposed in [31], which extracts content features based on attributes and relations between the entities to build a learning to rank model. In [31] this method is evaluated in the task of movie recommendation, considering only a set of features specific to film entities extracted from Freebase[2]. In contrast to our memory-based methods, which are domain-agnostic and rely only on the user interactions in search logs, the extraction of features for PRM-KNN requires manual selection and supervision for each target type of entities. Hence, for comparison purposes, we limit our evaluation of PRM-KNN to the task of movie recommendations as in [31], using the same features from that paper in a subset of our data that consists of clicks to movies. In addition to the general case, we provide the results of our best performing document-based approach on the same movie test set.

Regarding the metrics, we report Mean Reciprocal Rank (MRR) and P@1 over, e.g. P@5, since the average number of relevant items per session in the test set is 1.013, although F@5 is also provided. In addition to ranking quality measures, we report session coverage as the fraction of test sessions for which a given algorithm is able to compute recommendations.

Finally, we remark the difficulty of the recommendation task we are addressing. With an average of 1.013 relevant entities per session, the algorithms are required to find a single entity among a set of over 300k. Moreover, in order to simulate a real-world application, each method only computes a list of the top 10 recommended items. Relevant items ranked below that position are considered by the metrics as not retrieved at all.

[2]http://www.freebase.com

Table 1: Performance of the different variations of the document-based approach using dwell time to estimate $p(d|q,s)$. Best values highlighted in bold.

Similarity	$p(q\|s)$	MRR	P@1	F@5	Cov.
Users	Uniform	0.0811	0.0545	0.0389	99%
	Temporal	0.0888	0.0604	0.0422	99%
Queries	Uniform	0.0513	0.0368	0.0096	74%
	Temporal	0.0536	0.0393	0.0102	74%
Sessions	Uniform	0.0865	0.0570	0.0417	99%
	Temporal	**0.0936**	**0.0626**	**0.0447**	**99%**

4.3 Recommendation performance

We now discuss the performance of the proposed approaches for personalized entity recommendation. First, we present the results of the different variations from each approach separately, and then we compare the best of them against each other and against the baselines.

4.3.1 Document-based

Table 1 contains the results of the document-based methods with different combinations of document similarity and query relevance probability. The performance difference between the three variations of $p(d|q,s)$ was very small, but the method using the dwell time consistently achieved the best results. In Table 1 we report the results using the dwell time. We see that the *temporal* scheme for $p(q|s)$ consistently achieves the best performance. This confirms that queries and entities clicked later in the session are more useful to predict which entities are clicked next. The intuition for this is that entities clicked in short lapses of time are more likely to be related to the same task, since users' tasks and goals usually evolve during a search session.

Regarding the choice of similarity, we find that computing the similarity of entities by means of how often they are used within the same session is the most effective method, as opposed to users and queries. We argue that this is due to sessions being more specific than complete user profiles and query results, which allow to capture finer-grained correlations. It is worth noticing the nearly complete coverage of the document-based approach with user and session similarities. The coverage drops to 74% for queries, likely because it is more common for two entities to co-occur within a session or a user profile than within the results of a query, which are most of the time static.

4.3.2 Query-based

First, we analyzed the performance of each query selection strategy individually, that is, we tested if the *similar query, likely query* and *linked query* methods are able to correctly predict the next query the user will type. It is worth noticing the difference with respect to the main task addressed in this paper, which is predicting which entities will be clicked rather than queries. Unlike entities, that form a somewhat static set of candidate items for recommendation, the set of queries is much larger and dynamic. Moreover, according to our evaluation methodology the ground truth only contains

Table 2: Performance of the different variations of the query-based approach using dwell time to estimate $p(d|q,s)$ and temporal $p(q|s)$ whenever possible.

Method		MRR	P@1	F@5	Cov.
Similar query	Users	0.0545	0.0412	**0.0040**	78%
	Sessions	**0.0600**	**0.0455**	0.0039	**78%**
Likely query	Uniform	0.0438	0.0392	0.0007	38%
	Popularity	0.0443	0.0392	0.0009	42%
Linked query	Users	0.0399	0.0354	0.0001	53%
	Queries	0.0460	0.0414	0.0001	39%
	Sessions	0.0424	0.0375	0.0001	53%

Table 3: Performance of the session-based methods using dwell time for $p(d|q,s)$ and *temporal* $p(q|s)$.

Method	Items	MRR	P@1	F@5	Cov.
Jaccard	Docs.	0.0348	0.0213	0.0145	31%
	Queries	0.0438	0.0297	0.0165	37%
Centroid	Docs.	0.0335	0.0232	0.0112	**94%**
	Queries	0.0266	0.0173	0.0121	66%
IBM-1	Docs.	0.0311	0.0191	0.0136	2%
	Queries	0.0578	0.0432	0.0234	1%
WMS	Docs.	0.0583	0.0376	0.0249	2%
	Queries	**0.0656**	**0.0490**	**0.0270**	1%

the last query from each session and its clicked entities. We argue that this is a very challenging problem, as the algorithms have to find the single relevant query among a set of over six million. Furthermore, collaborative filtering techniques are known to suffer from extreme data sparsity [11] which would be exactly the case in this scenario. Due to lack of space we do not report the results here, but we confirmed this fact in our analysis, where the best performing method (*similar query* using sessions) achieved roughly 3% P@1.

The results for the entity recommendation experiments are shown in Table 2. We only report the performance of methods that use dwell time to measure within-session entity relevance, as the results with *clicks* and *rank* were very close. The first observation is that even though the methods were not successful in predicting the users' queries, they perform better when recommending entities. This seems to indicate that the queries retrieved in the first step may not match the user's query, but sometimes lead to useful entities. We also see that the *similar query* method achieves both the best accuracy and coverage, with a slight improvement when query similarity is defined based on session co-occurrence over users. Again, we argue that sessions being more specific allow to capture finer-grained correlations.

4.3.3 Session-based

Table 3 contains the results for the tested variations of the session-based approach. For the sake of space we report only values for the methods that use dwell time and *temporal* estimators for entity and query importance, respectively, as they yield the best results. The items column indicates which interactions were used to compute session similarities. We see that in most of the cases (except for the *centroid* method) we obtain better results using queries instead of documents (entities) to compute session similarities, at the expense of lower coverage. Since the number of queries is much larger than that of documents, the fact that two sessions share common queries is a stronger signal that they are related than sharing a document. On the other hand, given that the query space is so large, the probability that two sessions overlap is lower, resulting in degraded coverage. Regarding which similarity metric performs best, it is worth noticing the very low coverage of IBM-1 and WMS. This is a consequence of using LSH for finding only the top ten approximate nearest neighbors, which makes the training substantially faster but decreases the likelihood that two sessions share similar entities (respectively queries). Jaccard and Centroid similarities offer slightly worse accuracy but reasonable coverage. In particular, the Centroid method based on documents provides the best trade-off between accuracy and 94% of coverage.

Table 4: Comparison of the proposed approaches. Best values shown in bold. Statistically significant differences (t-test, $p < 0.001$) with respect to Next entity and PRM-KNN are marked in each case with $\uparrow$ if better and $\downarrow$ if worse.

Method	MRR	P@1	F@5	Cov.
Most popular	0.0027	0.0022	0.0011	**100%**
Daily popular	0.0162	0.0091	0.0084	**100%**
Next entity	0.0565	0.0468	0.0039	46%
Item kNN	0.0046	0.0020	0.0026	21%
Document-based	**0.0936**$\uparrow$	**0.0626**$\uparrow$	**0.0447**$\uparrow$	99%
Query-based	0.0600$\uparrow$	0.0455$\downarrow$	0.0039	78%
Session-based	0.0335$\downarrow$	0.0232$\downarrow$	0.0112$\uparrow$	94%
PRM-KNN (movies)	0.0664	0.0375	0.0316	25%
Doc.-based (movies)	0.1155$\uparrow$	0.0883$\uparrow$	0.0484$\uparrow$	30%

4.4 Comparison

In this section we compare the best performing methods from each of the proposed approaches against the baselines described in Section 4.2. The best performing document- and query-based methods are those with highest MRR in Tables 1 and 2, respectively. For the session-based we choose the Centroid similarity method using documents, as it provides the best trade-off between ranking accuracy and coverage. The results of the comparison are shown in Table 4. We see that *next entity* achieves the best performance among the simple baselines. However, this method cannot compute recommendations when the user's last click in the session corresponds to an entity that is not followed by any other in the search logs, which frequently happens to unpopular entities in the long tail. In summary, *next entity* is unable to recommend for users performing very specific or obscure tasks, which results in lower coverage. We also note the poor performance of Item kNN, which exploits the user's full history to compute personalized recommendations. As a consequence, this method is unable to compute recommendations for new users for which the system does not have any previous record. Furthermore, it does not take into account that short-term preferences are more relevant than long-term preferences to predict future clicks on entities for the target session.

Among our proposed approaches the document-based is performing best with respect to all metrics, with and im-

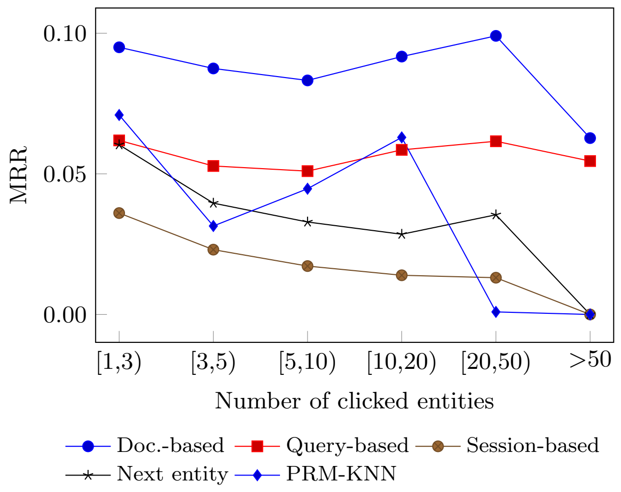

Figure 1: Average MRR for sessions of different lengths.

provement of 65% over the *next entity* baseline, followed by the query-based approach. Rather than looking at one-step click correlations, our methods take into account full search sessions to better capture the relative weight of each click within the session and to compute more meaningful similarities. On the other hand, the session-based approach is not as effective in finding similar sessions for recommendation using the centroids. Better accuracy can be achieved with WMS, which takes into account the importance of each document/query when computing the similarities, but at the cost of much lower coverage. Comparing the document- and session-based approaches, we conclude that in general it is better to recommend entities that are directly similar to those in the session rather than first finding similar sessions and suggesting their entities. Finally, we compare our best document-based approach against the PRM-KNN baseline for movie recommendations, using the settings described in [31] in a subset of our data. As previously discussed, we restrict our analysis of PRM-KNN to this subset because other types of entities would each require manually selecting and extracting new sets of features, making the comparison of our results against [31] less straightforward. The document-based approach performs significantly better and achieves superior coverage, and we note that the MRR values reported in [31] are considerably higher (0.451). We argue that this is due to our different evaluation methodology, in which only entities from future queries in the session are considered as ground truth. Furthermore, the discrepancy in our results could be explained by the difference in the datasets and the fact that we rely on Wikipedia instead of Freebase as a source of knowledge.

We also investigate how the quality of the recommendations depends on the amount of available preferences (clicks) in the input session. We split the sessions in the test set based on the number of clicked entities, and separately compute personalized recommendations for each group. Figure 1 shows the performance of each algorithm for different session length. As stated previously, the document-based approach outperforms every other method. The figure also shows an unexpected behavior. Instead of improving as more data is available in the session, the performance is degraded for the first few additional clicks. Then, as the user keeps browsing more entities, PRM-KNN and the document- and query-based methods start improving again, until the user clicks on about 20-50 entities. After that point the quality of recommendations drops again, although there are very few sessions with so many clicks. The previous behavior seems to indicate two types of successful scenarios for recommendation. In the first one the user has clicked only one or two entities in the session, and the relevant candidates to recommend are those frequently clicked next in the search logs. Methods based on co-occurrence of entities are able to capture this local correlation, which in turn leads to useful recommendations for short tasks. In this case, more data from previous queries in the session may confuse the algorithms by introducing noise, specially if the user is performing more than one information seeking task. In the second scenario, the user is engaged in a complex exploratory task that involves more entities. The system makes use of previous clicked entities to better understand the user's information goal, in order to find documents related to the whole task, rather than just the last clicked entity. However, as the session gets larger and more specific it becomes more difficult to find similar sessions in the logs, which explains the lower performance of the session-based approach for more complex tasks.

5. CROSS-DOMAIN RECOMMENDATIONS

Recommending of entities of *any* type in each search session is a very challenging problem. As we show in this paper, the performance of domain-agnostic CF-based recommender systems for entities is limited by the extreme sparsity of the data involved and by the amount of information available in the target session. However, in some applications the goal of the system is to provide suggestions of entities of a *specific* domain, e.g. movies. Furthermore, the system could analyze the type of entities clicked in the current session to decide which types the user is likely interested in. In these cases, the number of candidate items is significantly smaller than in the general case, which alleviates the sparsity and helps CF methods.

Hence, we turn our attention to a *cross-domain recommendation* scenario [7]. We aim to understand how the performance of entity recommendation methods depends on the domain of the target entities, and how it is also influenced by the domain of the available clicked entities in the current session. For this purpose, we classify the entities in our dataset in different domains as follows. First, we query DBpedia to extract the type of each entity by looking at the `rdf:type` property. Second, we mine DBpedia's ontology to extract the type hierarchy based on the `rdfs:subClassOf` property. Finally, we define domains based on root types and their descendants in the type hierarchy. For our analysis we considered the five domains spanned from the following types: `dbo:Book`, `dbo:Film`, `dbo:MusicalArtist`, `dbo:Person`, and `dbo:Place`. We assume an entity e belongs to domain $\mathcal{D}$ if its type matches $\mathcal{D}$ or either of its subtypes. Figure 2 shows the fraction of entities of each type in our dataset. We see that focusing on a particular domain the recommendation space is drastically reduced (e.g. by 97.3% in movies).

We select the best performing entity recommendation algorithm from the previous sections for our analysis, the document-based method that uses sessions for entity sim-

Table 5: MRR and coverage of the best document-based approach in several cross-domain recommendation scenarios. Values in bold are the best in each column (target domain). Statistically significant differences (t-test, $p < 0.01$) with respect each single-domain method are marked with ↑ if better and ↓ if worse.

	MRR						Coverage					
Source	All	Books	Movies	Music	People	Places	All	Books	Movies	Music	People	Places
All	0.0936	0.2305	0.1414↑	0.1858↓	0.1403↓	0.1046↓	99.0%	**79.5%**	**92.8%**	**93.9%**	**98.0%**	**93.2%**
All target		**0.2545**	0.1314↑	**0.2225↑**	**0.1572↑**	**0.1216**		15.1%	30.5%	37.3%	70.0%	50.0%
Books		0.2453	**0.2795↑**	0.0066↓	0.0924↓	0.0410↓		15.1%	3.7%	0.4%	1.0%	0.5%
Movies		0.2244	0.1155	0.0483↓	0.0771↓	0.0566↓		10.4%	30.5%	3.1%	5.4%	2.0%
Music		0.0092↓	0.0477↓	0.2111	0.1292↓	0.0187↓		1.8%	4.0%	37.3%	10.6%	2.1%
People		0.1627↓	0.0963↓	0.1907↓	0.1520	0.0540↓		26.7%	36.9%	65.5%	70.0%	16.1%
Places		0.0964↓	0.0526↓	0.0139↓	0.0189↓	0.1191		1.6%	2.1%	1.5%	2.5%	50.0%

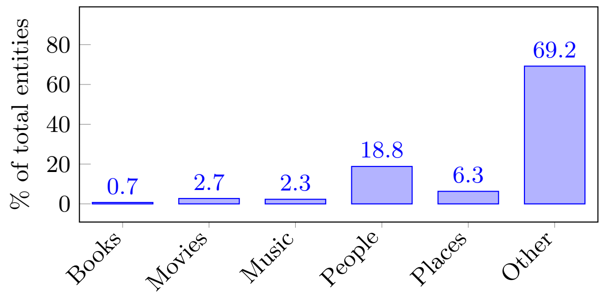

Figure 2: Distribution of domain entities in our dataset.

ilarity, dwell time as indicator of click relevance, and the *temporal* approach to favor the last clicked entities in the session. We compare the accuracy of the recommendations for several combinations of source and target domains. In this case *source domain* refers to the types of entities considered as input to the recommender system, which is still trained using all the data regardless of the target domain. The results are shown in Table 5. We first note the significant improvements in performance when recommendations are delivered in a specific target domain. This confirms our intuition that developing a universal entity recommender is a much harder problem than building a domain-focused system. Second, we see the drop in coverage when the source entities are restricted to a particular domain. Our goal is to understand if it is beneficial to consider entities from source domains different to the target, (i.e. to see if using *All* source entities is better than only using other *Books* when providing book recommendations) or if, on the other hand, cross-domain information is just introducing noise. Therefore, we also report the performance of the *All target* method, which is identical to *All* except its MRR is averaged to match the coverage of the single-domain method that uses source and target entities from the same domain.

Comparing the values of *All target* and the matching source for each target domain we see that in all cases the best performance is obtained using cross-domain information. Moreover, we observe interesting results when the source and target domains are completely different. For instance, it is possible to achieve good movie recommendations if the user only browsed books in the session, although only in those cases in which the system is able to compute predictions. The opposite also holds: movie entities are useful for predicting clicks on books. Respectively, clicks on entities related to *people* can also be used to predict which *books, music*, and to some extent *movies*, will the user browse. *People* entities are the most abundant in our dataset (see Figure 2), and are likely correlated with the previous domains through writers, music artists, and actors/actresses. On the other hand, *place* entities seem only useful for predicting other places.

Based on our study, we conclude that recommending entities from a specific domain is a considerably easier task than cross-domain recommendation, and that in this case the best performance can be achieved using cross-domain click data. We also confirm our intuition that some domains are more related than others and therefore more suitable for cross-domain recommendation. In this context, we hypothesize that the results of our analysis could be used to develop hybrid universal recommendation systems that integrate domain-specific subsystems, switching between them based on the user's observed click feedback.

6. CONCLUSIONS

Providing personalized entity recommendations to search engine users is a novel and challenging task. In this paper we have proposed three approaches inspired in Collaborative Filtering to predict which entities the user is likely to click next in the session, namely document-based, query-based, and session-based. We conducted extensive experiments to benchmark our methods using a dataset extracted from a large scale commercial search engine, and found that the document-based approach is able to outperform the best baseline by 65% while maintaining comparable coverage. We do not focus on a specific target domain (e.g. movies) as our methods do not make use of content information and are suitable for any type of entities. Finally, we analyzed the performance of our methods on a cross-domain scenario, and conclude that it is beneficial to not only consider user clicks of entities in the target domain, but also from related source domains.

One aspect that we did not address in this work is that of extracting task-oriented sessions from the search logs. In our experiments we sessionized the data based on a 30 minute time-out, but segmenting sessions according to tasks was shown to be better in personalized search [28]. Our methods need to compute pairwise similarities between entities and need to be retrained when new entities are added to the dataset. This may be a limitation in applications where the set of candidate entities for recommendation is dynamic, such as episodes from TV shows currently running.

Finally, we note that a potential benefit of a system capable of delivering personalized recommendations is to provide suggestions of entities that the user may be completely unaware of and therefore not able to find without assistance. A user study is in order to test if our methods are able to provide such serendipitous yet relevant recommendations and whether they actually increase user satisfaction.

7. ACKNOWLEDGMENTS

This work was supported by the Spanish Ministry of Science and Innovation (TIN2013-47090-C3-2).

8. REFERENCES

[1] E. Amitay, D. Carmel, N. Har'El, S. Ofek-Koifman, A. Soffer, S. Yogev, and N. Golbandi. Social search and discovery using a unified approach. In *Proc. of Hypertext 2009*, pages 199–208.

[2] A. Andoni and P. Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. *Commun. ACM*, 51(1):117–122, 2008.

[3] A. L. Berger and J. D. Lafferty. Information retrieval as statistical translation. In *Proc. of SIGIR 1999*, pages 222–229.

[4] B. Bi, H. Ma, B. P. Hsu, W. Chu, K. Wang, and J. Cho. Learning to recommend related entities to search users. In *Proc. of WSDM 2015*, pages 139–148.

[5] R. Blanco, B. B. Cambazoglu, P. Mika, and N. Torzec. Entity recommendations in web search. In *Proc. of ISWC 2013*, pages 33–48.

[6] R. Blanco, G. Ottaviano, and E. Meij. Fast and space-efficient entity linking for queries. In *Proc. of WSDM 2015*, pages 179–188.

[7] I. Cantador, I. Fernández-Tobías, S. Berkovsky, and P. Cremonesi. Cross-domain recommender systems. In *Recommender Systems Handbook*, pages 919–959. Springer US, 2015.

[8] B. Carterette, E. Kanoulas, and E. Yilmaz. Simulating simple user behavior for system effectiveness evaluation. In *Proc. of CIKM 2011*, pages 611–620.

[9] K. Chakrabarti, V. Ganti, J. Han, and D. Xin. Ranking objects based on relationships. In *Proc. of SIGMOD 2006*, pages 371–382.

[10] T. Cheng, X. Yan, and K. C.-C. Chang. Entityrank: Searching entities directly and holistically. In *Proc. of VLDB 2007*, pages 387–398.

[11] C. Desrosiers and G. Karypis. A comprehensive survey of neighborhood-based recommendation methods. In *Recommender Systems Handbook*, pages 107–144. Springer, 2011.

[12] L. Hollink, P. Mika, and R. Blanco. Web usage mining with semantic analysis. In *Proc. of WWW 2013*, pages 561–570.

[13] G. Jeh and J. Widom. Simrank: A measure of structural-context similarity. In *Proc. of KDD 2002*, pages 538–543.

[14] Y. Koren, S. C. North, and C. Volinsky. Measuring and extracting proximity graphs in networks. *ACM Trans. Knowl. Discov. Data*, 1(3), Dec. 2007.

[15] M. J. Kusner, Y. Sun, N. I. Kolkin, and K. Q. Weinberger. From word embeddings to document distances. In *Proc. of ICML 2015*, pages 957–966.

[16] G. Linden, B. Smith, and J. York. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing*, 7(1):76–80, 2003.

[17] P. Lops, M. de Gemmis, and G. Semeraro. Content-based recommender systems: State of the art and trends. In *Recommender Systems Handbook*, pages 73–105. Springer, 2011.

[18] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Proc. of NIPS 2013*, pages 3111–3119.

[19] I. Miliaraki, R. Blanco, and M. Lalmas. From selena gomez to marlon brando: Understanding explorative entity search. In *Proc. of WWW 2015*, pages 765–775.

[20] J. Pound, P. Mika, and H. Zaragoza. Ad-hoc object retrieval in the web of data. In *Proc. of WWW 2010*, pages 771–780.

[21] K. Radinsky, K. Svore, S. Dumais, J. Teevan, A. Bocharov, and E. Horvitz. Modeling and predicting behavioral dynamics on the web. In *Proc. of WWW 2012*, pages 599–608.

[22] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. In *Proc. of WWW 2010*, pages 285–295.

[23] Y. Song, X. Shi, and X. Fu. Evaluating and predicting user engagement change with degraded search relevance. In *Proc. of WWW 2013*, pages 1213–1224.

[24] D. Sontag, K. Collins-Thompson, P. N. Bennett, R. W. White, S. T. Dumais, and B. Billerbeck. Probabilistic models for personalizing web search. In *Proc. of WSDM 2012*, pages 433–442.

[25] K. Sugiyama, K. Hatano, and M. Yoshikawa. Adaptive web search based on user profile constructed without any effort from users. In *Proc. of WWW 2004*, pages 675–684.

[26] J. Teevan, S. T. Dumais, and E. Horvitz. Personalizing search via automated analysis of interests and activities. In *Proc. of SIGIR 2005*, pages 449–456.

[27] R. W. White, P. N. Bennett, and S. T. Dumais. Predicting short-term interests using activity-based search context. In *Proc. of CIKM 2010*, pages 1009–1018.

[28] R. W. White, W. Chu, A. H. Awadallah, X. He, Y. Song, and H. Wang. Enhancing personalized search by mining and modeling task behavior. In *Proc. of WWW 2013*, pages 1411–1420.

[29] B. Xiang, D. Jiang, J. Pei, X. Sun, E. Chen, and H. Li. Context-aware ranking in web search. In *Proc. of SIGIR 2010*, pages 451–458.

[30] S. Yogev, H. Roitman, D. Carmel, and N. Zwerdling. Towards expressive exploratory search over entity-relationship data. In *Proc. of WWW 2012*, pages 83–92.

[31] X. Yu, H. Ma, B.-J. P. Hsu, and J. Han. On building entity recommender systems using user click log and freebase knowledge. In *Proc. of WSDM 2014*, pages 263–272.