Ranking Entities Using Web Search Query Logs

Bodo Billerbeck[1], Gianluca Demartini[2],
Claudiu S. Firan[2], Tereza Iofciu[2], and Ralf Krestel[2]

[1] Microsoft Research, 7 JJ Thomson Avenue, Cambridge CB3 0FB, UK
`bodob@microsoft.com`
[2] L3S Research Center, Appelstr. 9a, 30167 Hannover, Germany
{`iofciu,demartini,firan,krestel`}`@L3S.de`

Abstract Searching for entities is an emerging task in Information Retrieval for which the goal is finding well defined entities instead of documents matching the query terms. In this paper we propose a novel approach to Entity Retrieval by using Web search engine query logs. We use Markov random walks on (1) Click Graphs – built from clickthrough data – and on (2) Session Graphs – built from user session information. We thus provide semantic bridges between different query terms, and therefore indicate meaningful connections between Entity Retrieval queries and related entities.

1 Introduction

Current Web search engines retrieve textually relevant Web pages for a given keyword query. The idea behind *Entity Retrieval* (ER) is to find entities directly. As an example, consider the ER query "hybrid cars" where relevant results would be *Toyota Prius* or *Honda Insight*, but not an informative page about hybrid vehicles. Instead of the user browsing through all Web pages retrieved by the search engine, a list of relevant entities should be presented to the user. As shown in previous work, a big percentage of web search engine queries are about entities [12]. A commercial product addressing such type of queries is Google Squared[3] where the results for queries such as "hybrid cars" is a table with instances of the desired type.

By mining a very large Web search engine query log with clickthrough data and session information we are able to create two types of graphs on which we can afterwards apply our algorithms: *(1)* We create a *Click Graph* by using queries and URLs as nodes and connecting and weighting them by their user click frequencies, and *(2)* A *Session Graph* by using only queries as nodes with edges between them if they appear in the same user sessions, again weighted by co-occurrence frequencies. In order to utilize this information source for improving ER we perform a Markov random walk on the graphs. We employ graph traversal techniques with different weighting schemes in order to match result entities to given queries. Experimental results show that the intersection of the click graph and the session graph is the best evidence for answering ER queries when traversing the graphs.

[3] `http://www.google.com/squared`

2 Related Work

Finding entities on the Web is a recent topic in the field of Information Retrieval. The first proposed approaches [3,4] mainly focus on scaling efficiently on Web dimension datasets but not on the effectiveness of search. In more detail, the authors of [4] tackle the ER task with a two component approach: one for extracting entities from the web and one for querying the database containing the extracted entities. A semantic search engine based on SPARQL queries, an optimized index structure, and an ontology is described in [3]. The system is implemented using YAGO([14]), a Wikipedia and WordNet based ontology, and Wikipedia itself as a corpus. The main differences of the above mentioned systems to our approach are that the user has to follow certain rules for querying the system; either stating the entity type that they are looking for or even some more complex structure requirements to transform the query into a SPARQL representation. We do not make any assumptions about the user query facilitating the interaction considerably. We also do not limit our system to certain entity types and use the Web as a corpus instead of e.g. Wikipedia.

In the wake of the INEX[4][8] challenge a couple of systems were presented to solve Entity Ranking in the Wikipedia context. Different strategies were used by the participants: The authors of [13] use link information on the Wikipedia pages; [10] make use of the category information present in Wikipedia and incorporate an ontology to improve effectiveness; [9] use NLP techniques; [15] leverages user provided example entities. A probabilistic framework for ER is proposed in [2].

Our ER algorithms exploit graph structures. Session Graphs or Click Graphs were previously used beneficially in various tasks. In [12] the authors perform an analysis of web search query logs and user activities concluding that 50% of queries are about entities. A probabilistic approach for named entity recognition in queries is presented in [11]. In [6] the authors describe how to use a Click Graph to improve Web search. In [7] session data is used to generate query suggestions. User session information is also used in [1] for improving Web search results. In our work we apply a Markov random walk model on both Click and Session Graphs and investigate its use for answering Entity Search tasks.

3 Constructing and Entering the Graphs

The Click Graph. A click log consists of a set of URLs $U = u_1, \ldots, u_n$ that users clicked on in response to queries $Q = q_1, \ldots, q_n$. Our approach for constructing the graphs is based on previous work of Craswell and Szummer [6]. We can build a *click graph* based on the notion of co-clicked URLs. In a click graph each unique query (i.e., a string of keywords) q_i and each URL u_j is a node. We define the set of nodes $V \equiv Q \cup U$. There is a directed edge between a query node q_i and a URL u_j if at least one user clicked u_j in the result page of the query q_i. Moreover, there is a weight on each edge computed based on the number of times u_j was clicked as result of query q_i. Such a graph represents relations between

[4] `http://www.inex.otago.ac.nz/`

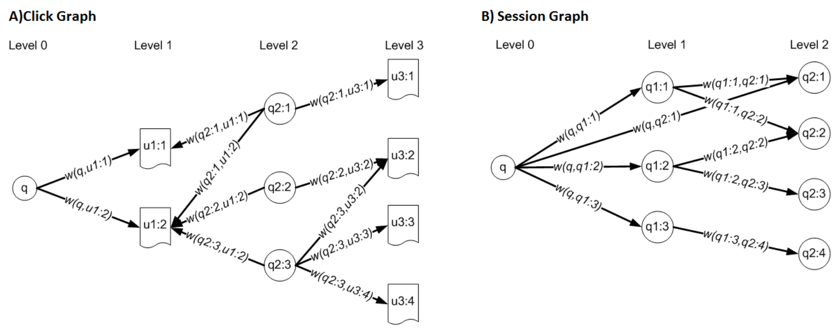

Figure 1. Schemes of a Click Graph (A), connecting an ER query q with entities $q_{l,i}$ via URLs $U_{l,j}$ where l indicates the level, and of a Session Graph (B) connecting a ER query q with queries $q_{2,i}$ on level 2.

queries and web documents as well as between different queries. We define q as the starting point for such search for entities: this is the ER query provided by the user (more details on how to properly select q are given in Section 3). We then assume queries close to q in the graph to be possible answers, that is, relevant entities q_i. In this way we can follow edges starting from q looking for relevant results (see Figure 1A).

The Session Graph. In a session graph nodes are formed by the set of queries $V \equiv Q = q_1, \ldots, q_n$. There is a directed link from a query q_i to a query q_j if the query q_j was issued after query q_i in the same search session. Similarly, we can define q as the starting point, that is, the user's ER query. We can then follow the edges looking for relevant results (that is, queries q_i) in the queries connected to q (see Figure 1B). Finally, the task of finding entities can be then defined as ranking queries q_i by probability of being relevant to the ER query q. The hypothesis is that a user posing an ER query which does not yield satisfying results will reformulate the query to find useful information. Upon inspection, it seems that the reformulated query often consists of an instance of the group of entities the user is looking for, e.g. "Spanish fish dishes" and "Paella".

Finding the Entry Point in the Graph. We investigate how we can identify a suitable subset of logged queries from which entities related to a particular topic can be extracted. We describe a possible way of selecting q (i.e., the starting point of the random walk) given the ER query issued by the user. We search the user query in the available query log and use such query as the node q. For instance, the query "salad recipes" can be found in the click graph as depicted in Figure 2. We then perform a random walk from this node in the graph. Beginning from this query, at the distance of two nodes out, the random walk finds such queries as "chicken salad recipe" as well as "pasta salad". Further out, the queries "green pea salad" and "caesar salad" are encountered. Specifically, we show the top ten queries with the highest transition probabilities from the node of origin (excluding the starting point), and a further five queries connected to two of these. While most of the queries directly linked to the original query are potentially useful for extracting entities, there are some queries that are less

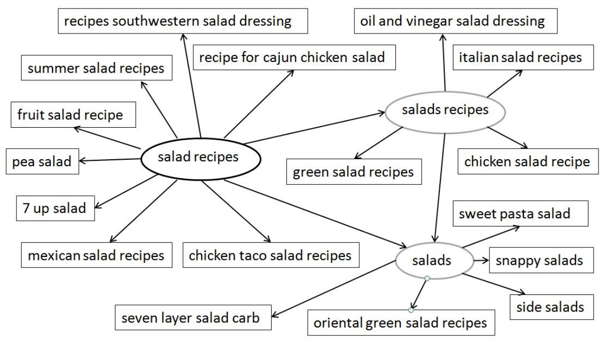

Figure 2. Selection of walked queries for the query "salad recipes".

suited for this task. However, these can be understood as categorising queries that may lead to other promising queries which may otherwise not be reached from the originating node. Examples of these 'bridging queries' are the nodes "salads" and "salads recipes" – singled out in Figure 2.

4 Walking the Graphs for Entity Ranking

Similarly to [6] we perform a Markov random walk on the click and session graphs in order to find relevant results for query q. The main difference is that our goal is to rank queries connected to q rather than ranking URLs by the probability distribution computed with the random walk. Moreover, the resulting entities are found only in the log queries, disregarding the text of the Web pages pointed to by the URLs in the log.

We define transition probabilities from a node j to a node k based on the click counts (i.e., $w(j,k)$ in Figure 1 A and B) as $P_{t+1|t}(k|j) = \frac{w(j,k)}{\sum_i w(j,i)}$ where i ranges over all nodes connected to j. The notation $P_{t+1|t}(k|j)$ defines the probability of moving from node j at step t to node k at step $t+1$.

By storing these single step transition probabilities in a matrix A where $A[j,k] = P_{t+1|t}(k|j)$, it is possible to compute a random walk of t steps starting from node j ($P_{t|0}(k|j)$) as $[A^t]_{jk}$. That is, we sum weights on the edges encountered on all paths of length t between the node j and a node k. The more paths the higher the random walk probability of reaching k starting from j.

4.1 Approaches Used on the Click Graph and Session Graph

At search time, the given ER query is matched in the graph and set as starting node (see Section 3). Performing a random walk over the graph, using query-URL-query transitions associated with weights on the edges (i.e. click frequencies), as shown in Figure 1A, enables us to find relevant entities as other queries in the graph and present them as a ranked list of entity results. We retrieve all queries reached within up to ten random walk steps in the click graph (i.e. five queries deep) and five steps in the session graph from the original query. The retrieved set of results is ranked and/or filtered by one of the following methods and only results appearing two steps away (i.e. one query deep) from the original query are kept as precision values drop rapidly when considering more levels.

Simple Random Walk. This approach ranks all reached queries (interpreted as potential entities) by their random walk probability computed as described in Section 4, (using 0 as self transition probability and only forward walks) but keeps only queries which are one URL away from the original query (i.e., level 2 in Figure 1A) for the method labelled C_2. For the method labelled C_{10}, we keep any queries encountered up to 10 steps away from the original queries. The result queries (potential entities) are ranked by their random walk transition scores over all possible paths up to the respective depth. $C_2\_rein_{10}$ is a hybrid of these two, only keeping queries at level 2, but the probability estimates are derived by walks of up to 10 steps into the click graph.

Clustered Results. The $C_2\_cluster$ method works similar to C_2 but scores are determined solely by the probabilities of moving from each query to any of the adjacent URLs. Queries at level 2 are clustered based on their co-clicked urls. Each such URL has a score based on clicks from level 2 queries. The URL score is then added to the scores of its level 2 queries. Starting from the graph formalization in Section 3, we can define the scores for a level 1 or 3 URL u_i based on the click counts from level 2 queries as $S_{url}(u_i) = \sum_j \frac{C(q_j,u_i)}{\sum_k C(q_j,u_k)}$ where j ranges over all the queries for which u_i was clicked and k ranges over all URLs connected to the query q_j. Level 2 query scores are then computed as $S_{query}(q_j) = \sum_i S_{url}(u_i)$ where u_i are all the clicked URLs for query q_j. For example, in Figure 1A, the score of $q_{2,2}$ would be a sum of the scores of its URLs, $u_{1,2}$ and $u_{3,2}$ (where $u_{1,2}$'s score is the average of clicks from $q_{2,1}$, $q_{2,2}$ and $q_{2,3}$).

Loops in the Graph. $C_2\_loop_{10}$ differs from C_2 by keeping only queries which can be reached via multiple paths starting from the given ER query (i.e., those that are connected via URLs at deeper levels, in this case up to 10 steps). This approach would keep only $q_{2,2}$ and $q_{2,3}$ in Figure 1A. A level 2 query q_i is only considered if the path after ten steps from the origin goes through a different level 2 query and comes back to the query q_i. This approach still uses the computed probability distribution to rank entities but limits the retrieved set to those well connected in the click graph. Therefore, the queries ranked for $C_2\_loop_{10}$ are a strict subset of those ranked for C_2, following the same ordering.

Simple Random Walk on the Session Graph. We perform a random walk over the session graph starting from a given ER query up to 5 steps away. Please note that 1 step on the Session Graph is equivalent to 2 steps on the Click Graph, where every other step ends on a URL, rather than a query. Considering the Session Graph we compared the following approaches for ranking entities. S_5: Starting from the original query (the ER query), walk to all queries reachable in 5 steps and rank them by their random walk probability as described in [6]. Analogous, S_1 ranks all the reached queries by their random walk probability when the random walk is performed on the first level only. That is, it does not explore the session graph at queries further away than those directly connected to the starting query. In Figure 1B, these would be the queries depicted on Level 1. Analoguously to $C_2\_rein_{10}$, $S_1\_rein_5$ forms a hybrid method.

4.2 Combining Click Graph Results with Session Graph Results

In order to exploit the two different graphs for answering the same query we can also use data fusion approaches given the two obtained rankings. In this paper we follow the simple approach of summing retrieval status values (RSVs) used for ranking entities for each approach[5] and normalizing them by the maximum score. In this way we combine scores computed with the click and session graph.

Union. As first approach, we unite the two sets of results retrieved from the click and session graphs. Their relevance scores (i.e. random walk probabilities) are normalized for each of the two approaches and if a result item appears in both result lists, these scores are added. We label these approaches as $U_{C,S}$ e.g. U_{C_2,S_1} in the case of the union of C_2 and S_1.

Intersection. We also rank entities combining the results of the random walk on the two graphs by keeping only results which are retrieved by both approaches. Again, the relevance scores from the single approaches are normalized and then added together. Such approaches are labelled as $I_{C,S}$ e.g. I_{C_2,S_1} for intersecting results from C_2 and S_1.

5 Evaluation

Experimental Setup. We use a query log from Bing[6]. It contains a sample of the most often clicked 35 million queries that were submitted over a period of 15 months by US American users to the search engine. This data consists of query as well as click specific details. Only query–URL pairs were retained for which at least 5 clicks were recorded overall. After some normalization of the queries there are 35 million unique queries and 44 million unique URLs. The session data consists of 25 million unique queries and a total 105 million unique query reformulations were recorded. For this purpose, we define a reformulation as two queries that were issued in the same search session within 10 minutes.

Ground Truth. In order to evaluate the proposed algorithms we constructed a benchmark for ER evaluation out of Wikipedia As gold standard we use the “List of” pages from Wikipedia. The title of such a page is used as an ER query (e.g., “lakes in Arizona”[7]). The titles of the Wikipedia pages that are linked to from such a “List of” page are considered to be relevant results (e.g., “Canyon Lake”, “Lake Havasu”, . . .). In order to use only queries that are more similar to typical Web queries in terms of length, we kept only those queries that consisted of 2 or 3 terms apart from “List of”. Thus we had 17,110 pages out of the total of 46,867 non-redirect “List of” pages. We matched these titles to queries in the log and kept the ones which appear at least 100 times in the query log and had at least 5 clicks on results. After this, we were left with 82 queries for evaluation.

Results. As a pre-processing step, all queries, both from the ground truth and from the query logs have had the stop words removed and were stemmed

[5] RSVs for ranking are the probabilities computed by the Markov Random Walk.

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

[7] `http://en.wikipedia.org/wiki/List_of_Arizona_lakes`

Method	**MAP**	**P@10**	**R-Prec**	**Queries Ranked**	**Relevant Entities Retrieved**
C_2	0.1423	0.0959	0.0541^{+}	489.54	8.79
S_1	0.1864	0.1026	0.1106*	78.61	6.87
$S_1\_rein_5$	0.2011*	0.1123	0.1082*	76.37	6.63
S_5	0.0252*$^{+}$	0.0768^{+}	0.0410^{+}	2454724.54	40.92
U_{C_2,S_1}	0.1438^{+}	0.1054	0.0792*	537.95	11.80
I_{C_2,S_1}	0.2285*	0.1146	0.1283*$^{+}$	29.13	2.78

Table 1. Results for finding entities using click and session graphs, averaged over the 82 ER queries in the evaluation set. Differences in MAP and R-Prec are statistically significant by means of Single Factor ANOVA. A * indicates statistical significant difference to C_2 and a $^+$ to S_1 (paired t-Test with $p <= 0.05$).

Method	**MAP**	**P@10**	**R-Prec**	**Queries Ranked**	**Relevant Entities Retrieved**
C_2	0.1423	0.0959	0.0541^{+}	489.54	8.79
$C_2\_cluster$	0.1490	0.1069*	0.0597*$^{+}$	489.72	8.79
$C_2\_loop_{10}$	0.1533*	0.1077*	0.0647*$^{+}$	358.16	8.45
$C_2\_rein_{10}$	0.1490	0.1069*	0.0597*$^{+}$	489.72	8.79
C_{10}	0.0548*$^{+}$	0.1	0.0549^{+}	87313.18	35.48

Table 2. Results for finding entities using click graphs. Statistical significance numbers are given to the same baselines in the previous table.

afterwards. We consider a retrieved entity to be relevant to an ER query if the string representing the relevant entity includes the ER query. In order to compare the different ranking approaches, we computed Mean Average Precision (MAP), precision for the first ten results (P@10) and R-Precision (R-Prec) of the produced rankings.

In Table 1 we compare our baseline runs C_2 and S_1 which are equivalent to ranking the queries directly connected to the user query by the weights on the edges. We can see that by using a Session Graph we obtain better results for ER queries. Moreover, while using the intersection of the Click and Session Graphs reduces the result set size significantly (29 results instead of 489 and 78 respectively), it improves effectiveness scores. With this simple approaches recall is anyway very low as the average number of relevant results per query is 83. The approach of unifying the sets of entities retrieved from the two graphs is not performing well mainly because of the large amount of retrieved entities.

In Table 2 we compare results of different approaches on the click graph (see Section 4.1). Our baseline is again C_2, that is a 2-steps random walk starting from the user query node, which is equivalent to ranking connected queries by the weights on the edges. We can see that a longer random walk (e.g., 10 steps away from the starting node, $C_2\_rein_{10}$) gives a better estimation of the relevance of level 2 queries. Moreover, we see that retrieving only queries that are also supported at deeper levels in a 10-step walk (i.e., $C_2\_loop_{10}$) improves the effectiveness. Here, most of the relevant entities retrieved are kept while on average more than 100 non-relevant are discarded.

6 Conclusions

We presented approaches for answering ER queries exploiting human behaviour stored in search engine query logs. After constructing click and session graphs out of the logs, we perform a Markov random walk on the graphs in order to rank queries which contain relevant entities to a given ER query. We created a gold standard out of Wikipedia "List Of" pages which can be reused for evaluating and comparing ER algorithms. Experimental results showed that integrating results from both the click and the session graph yields best effectiveness. Such results are promising as they would allow to build systems that, given a user ER query, can answer in real time with no need of highly complex algorithms.

Future work involves developing methods for grouping retrieved queries based on different similarity measures and extracting the core representative query for each group. This way, for an entity ranking query, we can present the results to the user as a short list of query representatives.

References

1. Eugene Agichtein, Eric Brill, and Susan Dumais. Improving web search ranking by incorporating user behavior information. In *SIGIR*, 2006.
2. K. Balog, M. Bron, and M. de Rijke. Category-based query modeling for entity search. In *ECIR*, 2010.
3. Holger Bast, Alexandru Chitea, Fabian Suchanek, and Ingmar Weber. Ester: efficient search on text, entities, and relations. In *SIGIR*, 2007.
4. Tao Cheng and Kevin Chen-Chuan Chang. Entity search engine: Towards agile best-effort information integration over the web. In *CIDR*, 2007.
5. Nick Craswell and Martin Szummer. Random walks on the click graph. In *SIGIR*, 2007.
6. Silviu Cucerzan and Ryen W. White. Query suggestion based on user landing pages. In *SIGIR*, 2007.
7. Arjen P. de Vries, Anne-Marie Vercoustre, James A. Thom, Nick Craswell, and Mounia Lalmas. Overview of the inex 2007 entity ranking track. In *INEX*, 2007.
8. Gianluca Demartini, Claudiu S. Firan, Tereza Iofciu, Ralf Krestel, and Wolfgang Nejdl. A model for ranking entities and its application to wikipedia. *LA-WEB*, 2008.
9. Gianluca Demartini, Claudiu S. Firan, Tereza Iofciu, and Wolfgang Nejdl. Semantically enhanced entity ranking. In *WISE*, 2008.
10. Jiafeng Guo, Gu Xu, Xueqi Cheng, and Hang Li. Named entity recognition in query. In *SIGIR*, 2009.
11. Ravi Kumar and Andrew Tomkins. A characterization of online search behavior. *IEEE Data Eng. Bull.*, 2009.
12. Jovan Pehcevski, Anne-Marie Vercoustre, and James A. Thom. Exploiting locality of wikipedia links in entity ranking. In *ECIR*, 2008.
13. Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. Yago: a core of semantic knowledge. In *WWW*, 2007.
14. Anne-Marie Vercoustre, James A. Thom, and Jovan Pehcevski. Entity ranking in wikipedia. In *SAC*, 2008.