Improving Ranking Consistency for Web Search by Leveraging a Knowledge Base and Search Logs

Jyun-Yu Jiang[†], Jing Liu[‡], Chin-Yew Lin[‡] and Pu-Jen Cheng[†]
[†]Department of Computer Science and Information Engineering, National Taiwan University
[‡]Microsoft Research
jyunyu.jiang@gmail.com, liudani@microsoft.com, cyl@microsoft.com, pjcheng@csie.ntu.edu.tw

ABSTRACT

In this paper, we propose a new idea called ranking consistency in web search. Relevance ranking is one of the biggest problems in creating an effective web search system. Given some queries with similar search intents, conventional approaches typically only optimize ranking models by each query separately. Hence, there are inconsistent rankings in modern search engines. It is expected that the search results of different queries with similar search intents should preserve ranking consistency. The aim of this paper is to learn consistent rankings in search results for improving the relevance ranking in web search. We then propose a re-ranking model aiming to simultaneously improve relevance ranking and ranking consistency by leveraging knowledge bases and search logs. To the best of our knowledge, our work offers the first solution to improving relevance rankings with ranking consistency. Extensive experiments have been conducted using the Freebase knowledge base and the large-scale query-log of a commercial search engine. The experimental results show that our approach significantly improves relevance ranking and ranking consistency. Two user surveys on Amazon Mechanical Turk also show that users are sensitive and prefer the consistent ranking results generated by our model.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval

Keywords

Web search; Ranking consistency; Re-ranking

1. INTRODUCTION

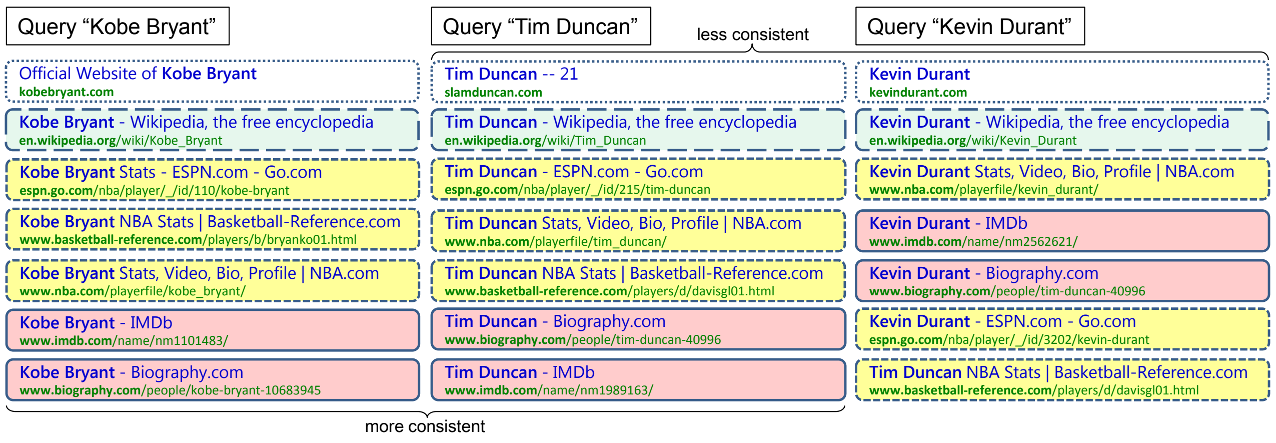
Relevance ranking is a critical task aiming to order search results by their relevance to a user's information needs. Previous works make great efforts to improve relevance rankings from different aspects such as the modeling of learning to rank [5,8,26], incorporating personalization [2,4,28] and the accurate evaluation of search results [12, 18, 29]. However, the ranking consistency[1] of search results for queries with similar search intents has not been carefully investigated. Suppose web pages about a certain topic on a website are treated as a topical cluster. Ranking consistency refers to the relevance of web pages in the same clusters that would be generally consistent in the search results for similar search intents.

Figure 1 shows some ordered search results for three queries about basketball players ("Kobe Bryant," "Tim Duncan," and "Kevin Durant") from a commercial search engine. In general, the ranking results of the first two queries ("Bryant" and "Duncan") are quite consistent by arranging `Wikipedia` at the top and ranking sports-related websites (e.g., `ESPN.com` and `NBA.com`) higher than others like movie-related ones (e.g., `IMDb`). According to our observation, most NBA players follow such consistency. Potentially, if one website focuses mainly on certain topics, the quality of their web pages is more likely to be close to each other. `ESPN.com` collects much personal information about sports-related players while `IMDb` focuses on movie promotion. For those queries with similar search intents, the relevance of these web pages about a certain topic (i.e., athletes) is likely to be consistent. In other words, `ESPN.com` is more relevant than `IMDb`. Unfortunately, the ranking of the third query ("Kevin Durant") is inconsistent in Figure 1. "Kevin Durant" is famous for basketball, rather than movies. Giving `IMDb` a higher ranking than `ESPN.com` does not make sense. The inconsistency represents a flaw in determining relevance ranking: each query is optimized separately so that the potential consistency among queries is totally ignored.

To learn the ranking consistency in web search, there are two challenges: (1) The first one is how to consistently rank topical clusters (i.e., web pages about a certain topic in each website) based on their relevance to those queries with similar search intents. For example, in Figure 1, we would like to discover that, for queries about basketball players, web pages in `ESPN.com` should be ranked higher than ones in `IMDb`. Then the web pages in topical clusters can be consistently ranked in the ranking results according to shared relevance. (2) The second challenge is how to generally rank all web pages for each query because some web pages (e.g., personal websites) may not share the topical cluster with web pages of other queries. Even though they cannot be ranked by the shared relevance with other web pages, they

CIKM'15, October 19–23, 2015, Melbourne, Australia.
© 2015 ACM. ISBN 978-1-4503-3794-6/15/10 ...$15.00.
DOI: http://dx.doi.org/10.1145/2806416.2806479.

[1]Although traditional approaches to learning to rank [8] also mention "consistency," their consistency is how the ranking results are consistent with the training orders. Different from their consistency, the ranking consistency in this paper is how the ranking results are consistent with the results of queries with similar search intents.



Query "Kobe Bryant"

Official Website of Kobe Bryant
kobebryant.com

Kobe Bryant - Wikipedia, the free encyclopedia
en.wikipedia.org/wiki/Kobe_Bryant

Kobe Bryant Stats - ESPN.com - Go.com
espn.go.com/nba/player/_/id/110/kobe-bryant

Kobe Bryant NBA Stats | Basketball-Reference.com
www.basketball-reference.com/players/b/bryanko01.html

Kobe Bryant Stats, Video, Bio, Profile | NBA.com
www.nba.com/playerfile/kobe_bryant/

Kobe Bryant - IMDb
www.imdb.com/name/nm1101483/

Kobe Bryant - Biography.com
www.biography.com/people/kobe-bryant-10683945

Query "Tim Duncan"

Tim Duncan -- 21
slamduncan.com

Tim Duncan - Wikipedia, the free encyclopedia
en.wikipedia.org/wiki/Tim_Duncan

Tim Duncan - ESPN.com - Go.com
espn.go.com/nba/player/_/id/215/tim-duncan

Tim Duncan Stats, Video, Bio, Profile | NBA.com
www.nba.com/playerfile/tim_duncan/

Tim Duncan NBA Stats | Basketball-Reference.com
www.basketball-reference.com/players/d/duncati01.html

Tim Duncan - Biography.com
www.biography.com/people/tim-duncan-40996

Tim Duncan - IMDb
www.imdb.com/name/nm1989163/

Query "Kevin Durant"

Kevin Durant
kevindurant.com

Kevin Durant - Wikipedia, the free encyclopedia
en.wikipedia.org/wiki/Kevin_Durant

Kevin Durant Stats, Video, Bio, Profile | NBA.com
www.nba.com/playerfile/kevin_durant/

Kevin Durant - IMDb
www.imdb.com/name/nm2562621/

Kevin Durant - Biography.com
www.biography.com/people/tim-duncan-40996

Kevin Durant - ESPN.com - Go.com
espn.go.com/nba/player/_/id/3202/kevin-durant

Tim Duncan NBA Stats | Basketball-Reference.com
www.basketball-reference.com/players/d/duncati01.html

Figure 1: Search results of three queries about NBA basketball player ("Kobe Bryant," "Tim Duncan" and "Kevin Durant") from a commercial search engine captured in April 2015.

are still relevant to specific queries. In Figure 1, three people all have personal websites with the highest relevance. Although these websites do not belong to any shared topical cluster[2], they should still be well ranked in proper positions. Even though sports-related websites are ranked higher than others, the order of different sports-related websites might be different for different queries. In Figure 1, the order of `NBA.com` and `Basketball-Reference.com` are reverse in the rankings of the first two queries. A possible reason is that the quality of contents in the two websites are different for the two queries. The second challenge also includes this issue.

In this paper, we propose a two-stage re-ranking model to conquer the above two challenges and to improve the relevance rankings of existing ranking models. In the first stage, we start with query type extraction for understanding the search intents of queries and URL pattern extraction for exploring topical clusters in each website. By exploiting click-through data in search logs as implicit feedback, numerous pairwise preferences of URL patterns are extracted for each query type, and then optimized as the complete relevance distribution by pairwise ranking aggregation. The rationality of this stage is also proven by an intuitive lemma. In the second stage, an ensemble-based re-ranking model is proposed to generally rank web pages by the results from the previous stage and the original ranker of an existing search engine. By applying click-through data as training data and incorporating eight useful features, the model finds an appropriate way to re-rank each web page from original results.

Extensive experiments have been conducted on the large-scale real-world search log from a commercial search engine. As this is the first study on the ranking consistency in web search, we propose a new metric to evaluate the consistency. The experimental results show that our model significantly improves not only the ranking consistency but also the relevance rankings from the original ranker across various subsets of testing data with different distribution and query types. Moreover, two user surveys on Amazon Mechanical Turk further show the effectiveness of the ranking consistency generated by our approach.

[2]In `ESPN.com`, the web pages share the same topic for various queries (e.g., basketball players). But the web pages in `kobebryant.com` are all about the same query (i.e., "Kobe Bryant").

The rest of this paper is organized as follows. After covering related work in Section 2, we give a brief overview of the two-stage re-ranking model in Section 3. The details of our model are presented in Sections 4 and 5. Section 6 reports the experimental results of our model and two user surveys on Amazon Mechanical Turk. Finally, Section 7 gives our discussions and conclusions.

2. RELATED WORK

In this section, we review substantial previous work related to this paper in a number of topics.

Ranking Consistency. Although ranking consistency has never been discussed in relation to web search, the term ranking consistency is used in other other fields. In image matching and object retrieval, previous works have evaluated ranking consistency by measuring the similarity between image contents [10]. However, such ranking consistency is only about the image search results of the same query. Moreover, this is adopted to estimate similarities between two ranking results of different methods for the same ranked items by Nguyen and Szymanski [25]. Nevertheless, they calculate similarities between the same objects and ranked items, but not between the search results of different queries representing similar topics. Our ranking consistency is calculated at a higher level and is able to have more general applications.

URL Patterns of Webpages. A URL pattern summarizes websites with similar URL strings and properties. This feature is an important characteristic for representing web pages in the deep web. Previous works [16, 20, 31] represent URL patterns as regular expression patterns so that URLs can be matched conveniently. Jiang *et al.* [16] and Li *et al.* [20] learn URL patterns from training URL collections for crawling web forums. Yin *et al.* [31] extract URL patterns from click-through data for information extraction. In our work, we collect URL collections from click-through data and learn URL regular expressions patterns as in [16]. By leveraging click-through data, we derive the user preferences for URL patterns. Therefore, web search performance and ranking consistency can improve if we rank websites consistent with preferences.

Search Intents behind Queries. Understanding users' search intents is an important task for search engines. Some previous work represent the intents with click-through data [21]. Hu *et al.* [15] use Wikipedia concepts to represent user search intents. Guo *et al.* [14] propose a new task called named entity recognition in query (NERQ). They find that more than 70% of queries contain entities [14]. Based on this analysis, in our work, we assume that the search intents of queries can be clustered into many types in a knowledge base, i.e. Freebase, according to entities contained in queries. Moreover, understanding search intents behind queries can also improve relevance ranking. Gao *et al.* [13] extract features from click graphs as additional features. Bennett *et al.* [3] adopt predefined classes and try to estimate class distribution as search intents. However, they do not consider the ranking consistency of search results for multiple queries. Gao *et al.* [13] only treat click-through information as search intents of queries. Bennett *et al.* [3] optimize ranking models with query and document class distribution for each query independently.

Click-through Data in Web Search For a search engine, click-through data is an informative relevance signal from users about relevance ranking quality [18]. Clicks in search engine logs have been used as implicit measures such as satisfied clicks (SAT-Clicks) for evaluation [12]. Learning from click-through data also helps improve the relevance ranking of search engines. Joachims [17] optimizes the search quality with implicit feedback extracted from click-through data. Radlinski and Joachims [26] generate user preferences from search logs with connected query chains. However, they consider only preferences between web pages in the search results of a single query, not between the URL patterns for different queries. If user preferences between URL patterns are also considered, the relevance of rare web pages can be better estimated by leveraging information from popular ones with similar topics.

Pairwise Ranking Aggregation. Our approach deriving consistent rankings of URL patterns is relevant to pairwise ranking aggregation. In web search, Carterette *et al.* [9] and Farah *et al.* [11] aggregate search results of a single query from multiple search engines. In other fields, Bennett *et al.* [1] improve image search quality by aggregating crowdsourcing pairwise annotations. Liu *et al.* [22] aggregate generated pairwise competitions to estimate user expertise in community question answering services. However, they apply ranking aggregation for a single query or only one problem. In this paper, we derive consistent rankings of URL patterns, and then apply to multiple queries.

To summarize the relationship to previous approaches, our approach (1) considers the search results of multiple queries at the same time for ranking consistency; (2) utilizes click-through data to learn URL patterns and represent websites in the deep web; (3) discovers queries with similar intents by leveraging type information in knowledge bases; and (4) exploits click-through data to find out users' preferences between URL patterns for each query type. To the best of our knowledge, this is the first study on ranking consistency in web search.

3. TWO-STAGE RE-RANKING MODEL

In this section, we first give a brief overview of the proposed approach, and then describe the details in the following sections. Based on the concept of the ranking consistency mentioned in Section 1, to simultaneously improve the ranking consistency and relevance rankings, we propose a re-ranking model consisting of two stages: (1) consistent ranking model and (2) ensemble-based re-ranking.

The first stage, i.e., consistent ranking model, is to consistently rank web pages in topical clusters by the relevance of clusters to queries with same types, which are defined as the groups of similar search intents. Hence, to develop the ranking consistency, there are two challenges before ranking web pages: (1) how to discover same-type queries (in other words, how to identify the query types); and (2) how to discover the topical clusters (i.e., web pages with similar topics) in each website.

To recognize the search intents behind queries, a possible solution is named entity recognition in queries (NERQ) [14]. It has been shown that the types of mentioned entities within a query can be used to represent query intents, and more than 70% of queries cover entities [14]. Moreover, many existing knowledge bases (e.g., Freebase [6] and Probase [30]) summarize the types of entities. For example, in Figure 1, the first query will lead to the entity "Kobe Bryant" with the types `people/person`, `basketball_player` and `film/actor`. Hence, the knowledge base can be leveraged to identify the types of the entities as the query types.

As some websites have specific contents (e.g., `ESPN.com` contains web pages about different sports), we need to discover the topical clusters of web pages in each website, where the web pages in each cluster share the same topic. The URL patterns can be utilized to represent the topical clusters because web pages with the same topic usually share the same URL pattern. For instance, in Figure 1, web pages of `ESPN.com` for three queries can be summarized as the URL pattern `espn.go.com/nba/player/_/id/*/*/`. Moreover, the utilization of URL patterns is helpful in determining the ranking consistency. Particularly, the ranking consistency of search results can be decided by the consistency of matched URL patterns. Note that personal websites such as `kobebryant.com` and `slamduncan.com` may not be matched by any URL pattern because their URLs are usually unique and dissimilar to other URLs.

To estimate the relevance of URL patterns (i.e., topical clusters) to query types, we apply the click-through data in search logs because clicks can represent, to some extent, users' satisfaction. By exploring implicit feedback from user clicks [17], numerous pairwise preferences of URL patterns are collected for each query type, and then aggregated into weighted preferences to avoid the bias of too popular queries and contradictions. With the preferences of URL patterns, the cost function of Rank-Net [8] is applied to combine such preferences and optimize the relevance of patterns as probability-like scores by stochastic gradient descent (SGD). In the example of Figure 1, if most users click web pages in `ESPN.com` but not the ones in `IMDb`, the URL pattern of `ESPN.com` will have a higher score than the pattern of `IMDb`.

The second stage, i.e., ensemble-based re-ranking, is to re-rank the ranking results of an original ranker based on the results from the consistent ranking model (Stage 1). The re-ranking model shapes a parameter to decide the weights of the original ranker and the consistent ranking model, and then makes an ensemble of two approaches. However, such parameters should be different for each query and each web page under various situations. For example, in Figure 1, personal websites like `kobebryant.com` are not matched by any URL pattern, so the weight of the original ranker would be larger to make them be ranked in higher positions. Conversely, web pages in `ESPN.com` are matched by a URL pattern, such that the weight would be smaller to gain relevance from the ranking consistency over the queries of the same type. To model various situations, eight features within three categories are proposed, and the parameter can be optimized as a logistic function whose inputs are the features. Finally, the re-ranking model can generally re-rank not only web pages in topical clusters but other web pages, and then well optimize the relevance rankings.

4. CONSISTENT RANKING MODEL

In this section, we discuss the details of the first stage of the proposed approach, which aims to rank web pages in topical clusters according their relevance to query types.

4.1 Model Formalities

Recall that in Section 3 we discuss ranking consistency according to queries of the same types and web pages matched by URL patterns. To apply URL patterns to relevance rankings, we first make an assumption as follows:

Assumption 1 *For the URL patterns that are more relevant to query types, web pages matched by such URL patterns may be also more relevant to the queries.*

With the above assumption, for those queries with only single type, a simple approach is to rank the web pages that are matched with more relevant patterns to higher positions and ignore those web pages without any matched pattern. It ensures consistent rankings because more relevant patterns must be ranked higher than less relevant patterns.

However, a query may have multiple types due to its ambiguity. Moreover, URL patterns may have different levels of relevance for each type. Based on the above approach, we start our model from the following assumption:

Assumption 2 *When a query belongs to multiple types, the relevance distribution is an aggregation over types.*

That is, under the assumption, the relevance distribution of a query can be computed by the distribution of its corresponding types. To model the problem more easily, we use a probability distribution to represent the relevance score. This assumption is natural for a type-based consistent ranking model. Considering all types of a query, the model is able to cover potentially esoteric search intents. Moreover, the problems can be decomposed into type-related tasks, which are simpler and more reliable.

Query Type Extraction. To extract the types of a query, we need to link the query to the corresponding entity in the knowledge base, then the types of the linked entity will be the query types. Although there has been some previous work about named entity recognition in query [14], as a pioneer study of the ranking consistency, we simply exploit the click-through data and make connections between queries and Wikipedia pages of entities. Note that the Wikipedia pages of entities can be found in most of knowledge bases such as Freebase [6]. If an entity's Wikipedia page has been clicked many times after users have submitted a query, the query will likely contain such an entity. Here an entity is treated as the intended entity of a query if the click-through rate of its Wikipedia page is more than 10%. After mapping queries to entities, we transfer types of entities in Freebase [6] to queries as their types.

URL Pattern Extraction. Previous works preset many methods for extracting URL patterns from click-through data [31] or URL collections [16]. In our work, we adopt the method mentioned in [16] because some URL patterns are so complex that we have to use regular expressions to represent them. All URLs clicked in the training logs are

1. Kobe Bryant - Wikipedia, the free encyclopedia
   en.wikipedia.org/wiki/Kobe_Bryant
   p_1 = `en.wikipedia.org/wiki/*`
2. **(Clicked) KB24 - Official Website of Kobe Bryant**
   kobebryant.com, $p_2 = \emptyset$
3. **(Clicked) Kobe Bryant Stats, Video, Bio, Profile**
   www.nba.com/playerfile/kobe_bryant/
   p_3 = `www.nba.com/playerfile/*/`
4. Kobe Bryant Stats, News, Videos, Highlights ...
   espn.go.com/nba/player/_/id/110/kobe-bryant
   p_4 = `espn.go.com/nba/player/_/id/*/*/`
5. Kobe Bryant Biography
   www.biography.com/people/kobe-bryant-10683945
   p_5 = `www.biography.com/people/*`
6. **(Clicked) Kobe Bryant | Los Angeles Lakers**
   sports.yahoo.com/nba/players/3118
   p_6 = `sports.yahoo.com/nba/players/*`

Figure 2: Ranking presented for the query "Kobe Bryant" about a famous basketball player. Five URLs are matched by URL patterns, and the user clicked on URLs 2, 3 and 6.

collected as a URL collection. We then generalize the URL collection into many regular expressions as URL patterns.

4.2 Model Formulation

Here our model is formulated formally. In this model, URLs are used to represent web pages. For a query q, we would like to estimate the relevance distribution $P(u \mid q)$, where u is a web page. In order to rank consistently, we only focus on URLs, which can be matched by URL patterns. So we have:

$$P(u \mid q) = \begin{cases} P(p \mid q) & \text{, if } u \text{ is matched by pattern } p \\ 0 & \text{, otherwise} \end{cases},$$

where $P(p \mid q)$ is the relevance distribution of pattern p given the query q. We ignore other URLs without any pattern-match and treat them as irrelevant in this stage. Furthermore, they will be well re-ranked in the second stage with several robust features.

Following Assumption 2, the model ranks a pattern p according to the probabilities that p is relevant to q's types $T(q)$ in the knowledge base. On the other hand, queries are independent to the relevance distribution of URL patterns. Hence, the relevance distribution $P(p \mid q)$ can be re-written by marginalization as follows:

$$\begin{aligned} P(p \mid q) &= \sum_{t \in T(q)} P(p \mid t, q) \cdot P(t \mid q) \\ &= \sum_{t \in T(q)} P(p \mid t) \cdot P(t \mid q) \end{aligned}$$

Therefore, the task can be decomposed into two problems with better reliability: (1) how to estimate the pattern-type relevance $P(p \mid t)$, and (2) how to estimate the type distribution $P(t \mid q)$.

4.3 Pattern-Type Relevance

To estimate the pattern-type relevance $P(p \mid t)$ in the real-world, we adopt a data-driven approach with click-through data from search engine logs. We extract pairwise preferences from original search results and user preference feedback from their behaviors in authentic web search scenarios.

Implicit Feedback from Clicks. As defined in [17], click-through data in search engines can be thought of as

Algorithm 1 Preference Extraction

```
Input: Given entity: $e$; Queries of $e$ in the training logs:
    $Q_e = \{(q, C)\}$, where $q$ is an entity query of $e$, and $C$
    is the set of SAT-clicked URLs in click-through data;
    Pattern set: $S$.
Output: Initial preference list for $e$: $D_e$.
 1: $D_e \leftarrow [\ ]$
 2: for $(q, C) \in Q_e$ do
 3:     $clickP \leftarrow \emptyset$
 4:     for $(u, p) \in C \times S$ do
 5:         if $u$ is matched by $p$ then
 6:             $clickP \leftarrow clickP \cup \{p\}$
 7:         end if
 8:     end for
 9:     for $(p_i, p_j) \in clickP \times (S - clickP)$ do
10:         Add $(p_i, p_j)$ into $D_e$
11:     end for
12: end for
13: return $D_e$
```

Algorithm 2 Preference Aggregation

```
Input: Given entity: $e$; Initial preference list of $e$: $D_e$.
Output: Weighted preference list for the entity $e$: $R_e$.
 1: $cnt \leftarrow$ new RB-Tree (or any key-value data structure)
 2: for $(p_i, p_j) \in D_e$ do
 3:     if $(p_i, p_j) \notin cnt$ then
 4:         $cnt[(p_i, p_j)] \leftarrow 0$
 5:         $cnt[(p_j, p_i)] \leftarrow 0$
 6:     end if
 7:     $cnt[(p_i, p_j)] \leftarrow cnt[(p_i, p_j)] + 1$
 8: end for
 9: $R_e \leftarrow [\ ]$
10: for $(p_i, p_j) \in S \times S$ do
11:     if $(p_i, p_j) \in cnt$ and $cnt[(p_i, p_j)] > 0$ then
12:         $w_{ij} \leftarrow cnt[(p_i, p_j)] \,/\, (cnt[(p_i, p_j)] + cnt[(p_j, p_i)])$
13:         Add $(p_i, p_j, w_{ij})$ into $R_e$
14:     end if
15: end for
16: return $R_e$
```

triplets (q, r, c) consisting of the query q, the ranking r presented to the user and the set c of clicked URLs. Figure 2 presents an example of click-through data: the user enters a query, receives the ranking from the search engine, and then clicks on several URLs with his or her opinions. Figure 2 also shows that some URLs can be matched by URL patterns. Here we only focus on these matched URLs.

Using click-through data, many previous works extract user preference feedback by assuming that a user is more likely to click on a URL which is more relevant to the query for her [17]. Here we make a similar assumption: a user is more likely to click a URL matched by a pattern if the pattern is more relevant to types of queries. Different from previous works which only considering the unclicked URLs ranked higher than clicked ones, we assume users observe the top k URLs and extract preferences from unclicked patterns, this is because: (1) we would also like to leverage information from the original rankers, not only users' preference feedback; and (2) if we do not generate preferences between patterns in lower ranks, there will be a bias to such patterns. Consider the example in Figure 2, the user enters a query about a basketball player and clicks the patterns p_3 and p_6. We can extract pairwise preferences as follows:

$$(p_3, p_1), (p_3, p_4), (p_3, p_5), (p_6, p_1), (p_6, p_4), (p_6, p_5).$$

Note that we do not consider p_2 because the URL is not matched by any URL pattern. For each preference (p_i, p_j), it shows that the pattern p_i is more relevant to the type `basketball_player` than p_j in this click-through data. In addition to patterns in the search results, we also assume clicked patterns are more satisfactory than patterns which do not appear in top k URLs in search results. This is because these patterns might have lower relevance to entities' types such that they cannot be ranked higher. In our work, we focus on top k URLs, which are contents of the first page in search results. Moreover, the queries might have no matched URLs. As in previous works [2,4], to ensure the reliability of clicks, we consider only URLs with satisfied clicks (SAT-Clicks) as relevant and treat the other results as irrelevant. To define a SAT-Click, we adopt the definition used in previous work [29] and consider clicks with at least 30 seconds dwell time as SAT-Clicks.

Based on the above strategies, Algorithm 1 extracts the initial preference list D_e. Therefore, we can collect the click-through data of queries with each type and extract preferences. However, there is a drawback. The results might be biased by queries of popular entities because we extract preferences from click-through data respectively. To avoid popularity bias, we aggregate preferences by entities of queries. It is more natural for estimating pattern-type relevance. If a pattern is actually more relevant to a type than the other, we expect that most entities with such a type might make corresponding preferences. Hence, Algorithm 2 aggregates the initial preferences D_e from Algorithm 1 as a weighted preference list R_e for each entity e. Note that we apply the red-black tree (RB-Tree) as the data structure to aggregate occurrences of preferences. More precisely, the physical meaning of weights w_{ij} in the weighted preference list is to estimate the probability of observing the preference for the two patterns. Finally, we have many weighted preferences for each type with better rationalities by aggregating original preferences.

Relevance Optimization. A pairwise preference can be thought of as a partial constraint of two patterns because it represents a pairwise comparison of them. With these preferences, we optimize the pattern-type relevance such that constraints can be satisfied.

First, we assume the distribution of the pattern-type relevance for the pattern p_i can be represented by a logistic function as follows:

$$P(p_i \mid t) = \frac{1}{1 + \exp(-\theta_{i,t})},$$

where $\theta_{i,t}$ is a parameter for p_i and the type t. Therefore, we can optimize $\theta_{i,t}$ to fit constraints from preferences. Specifically, we adopt a gradient descent method with the RankNet cost function [8] to optimize pairwise preference. For each type t, given the set of entities with such type $E(t)$, the RankNet cost function can be computed as follows:

$$\sum_{e \in E(t)} \sum_{(p_1, p_2, w) \in R_e} w \cdot \log\left(1 + \exp\left(P(p_2 \mid t) - P(p_1 \mid t)\right)\right),$$

where R_e is the list of weighted preferences for the entity e, and $P(p_i \mid t)$ is the relevance using current parameters $\theta_{i,t}$. We can then compute gradients for each parameter $\theta_{i,t}$ and minimize the cost function.

Hence, we obtain the pattern-type relevance $P(p_i \mid t)$ of pattern p_i for each type t from optimized parameters $\theta_{i,t}$.

4.4 Type Distribution

The type distribution of a query shows how much users treat it as a query of such a type. For example, although Kobe Bryant types "basketball player" and "actor" at the same time, people are much more concerned about the identity of basketball player. Accordingly, the URL pattern `www.nba.com/playerfile/*/` should be more relevant to Kobe Bryant than the pattern `www.imdb.com/name/*/`.

To estimate type distribution, we use click-through data again. During a search for the query, we assume that users are more likely to click URLs matched by patterns of types with which users are more concerned. Following the above assumption, we can estimate type distribution for a query q by a simple Bayesian m-estimate smoothing [24] as follows:

$$P(t \mid q) = \frac{\sum_{p \in S} P(t \mid p) \cdot Click(p,q) + m \cdot P(t)}{m + \sum_{t \in T(q)} \sum_{p \in S} P(t \mid p) \cdot Click(p,q)},$$

where S is the set containing all patterns. Here $Click(p,q)$ calculates the times users are clicking URLs matched by pattern p during searches for the query q. The prior distribution $P(t)$ can be computed by normalizing the number of entities for each type. The type distribution of patterns $P(t \mid p)$ can be calculated with the Bayes' theorem as follows:

$$P(t \mid p) = \frac{P(p \mid t) \cdot P(t)}{P(p)},$$

where $P(p \mid t)$ is pattern-type relevance estimated in Section 4.3. The pattern distribution $P(p)$ can be computed by normalizing the number of SAT-Clicks for each URL pattern in the pattern set S.

4.5 Theoretical Analysis

To verify the rationality of the consistent ranking model, we conduct a theoretical analysis. Recall that the model follows Assumption 1 and ranks pattern-matched URLs by pattern relevance. Here we consider only the URLs matched by patterns (i.e., non-matched URLs like personal sites are not analyzed here because they will be re-ranked in the second stage by appropriate features as shown in Section 5). Before the analysis, note that the effectiveness of a ranking pair (r_1, r_2) can be computed as the average of the effectiveness of two rankings (i.e., $F(r_1, r_2) = (F(r_1) + F(r_2))/2$). It is consistent with most traditional evaluation measures in information retrieval such as NDCG and MAP.

Consider the ideal case of Assumption 1 (i.e., the relevance of matched URL patterns perfectly leading to the relevance of web pages), we have the following lemma:

Lemma 1 *If the relevance of the web pages is exactly consistent with the relevance of matched URL patterns, for any pair of inconsistent rankings (r_1, r_2) of web pages matched by the identical set of URL patterns for two same-type queries, there must be a consistent pair (r_1', r_2') such that the consistent one is at least as effective as the inconsistent one.*

PROOF. Let the effectiveness of a ranking r be $F(r)$, e.g., reciprocal rank or average precision. We prove the theorem with a proof by contradiction. First, we start by assuming there is no ranking pair (r_1', r_2') such that:

$$F(r_1') + F(r_2') \geq F(r_1) + F(r_2),$$

where r_1' and r_2' are consistent.

However, we can construct the counterexample with a structured approach as follows. If $F(r_1) \geq F(r_2)$, let $r_1' = r_1$, and re-rank r_2 as r_2' such that the order of matched URL patterns is identical to r_1. Note that (r_1', r_2') must be consistent because both of them rank web pages identically with r_1. So we have:

$$F(r_1') + F(r_2') = F(r_1) + F(r_1) \geq F(r_1) + F(r_2),$$

which contradicts the initial assumption, then proving the theorem. If $F(r_1) < F(r_2)$, we can still find such a pair with similar methods. Hence, we are guaranteed to find a consistent ranking pair which is at least as effective as the inconsistent pair. □

Hence, the consistent ranking model is rational because, for pattern-matched web pages, the above lemma shows that to improve consistency with pattern relevance may benefit relevance rankings under Assumption 1.

5. ENSEMBLE-BASED RE-RANKING

In Section 4, we propose a method to rank pattern-matched URLs consistently. However, in real situations, there are many URLs without any matched patterns such as personal sites and official websites of entities. These URLs might be as relevant as or more relevant than pattern-matched URLs. Here we propose an ensemble-based re-ranking model to leverage information from both the original ranker and the proposed consistent ranking model.

5.1 Model Formulation

We start by presenting an ensemble-based model that assumes uniform contributions from the consistent ranking model and the original ranker. For a query q, we compute the relevance of a URL ranked in the position i by the original ranker as follows:

$$P(u \mid q, i) = \lambda \cdot P(u \mid q) + (1 - \lambda) \cdot P(u \mid i),$$

where $P(u \mid q)$ and $P(u \mid i)$ are relevance given by the consistent ranking model and the original ranker, λ is a parameter that controls the impact of the former model. In our work, we simply let $P(u \mid i)$ be a linear decay function $(k - i + 1)/k$, where k is the number of URLs in each page of search results.

5.2 Multiple Parameters

Using a single parameter is not always reasonable. In different cases, the most suitable parameter might be different. For example, when the URL is a personal site without any matched pattern, λ should be lower because it may be so relevant, and the consistent ranking model gives zero relevance. As in [27], we replace the parameter λ with a logistic function, which allows some features as the input, whereas the output is bounded to values between 0 and 1.

More formally, the input of a logistic function is a set of features X, whereas the output is the probability-like value predicted by fitting X to a logistic curve. Accordingly, the parameter λ is replaced as follows:

$$\lambda(X) = \frac{1}{1 + \exp(-f(X))}, \quad f(X) = \beta_0 + \sum_{i=1}^{|X|} \beta_i \cdot x_i,$$

where x_i is the i-th feature in X, β_i is the coefficient parameter of x_i, and β_0 is the bias parameter. Note that $|X| = 0$ is a special case of single parameter.

5.3 Parameter Optimization

In the proposed re-ranking model, we would like to learn β parameters for receiving better retrieval performance. As in

[23,27], we use a gradient descent method with the RankNet cost function [8] again to perform an effective parameter optimization. With the assumption mentioned in Section 4.3, we assume that clicked URLs are more relevant than unclicked ones. Therefore, we can extract many pairwise preferences of URLs for optimization. Given a set of queries Q, the cost function can be computed as follows:

$$\sum_{q_e \in Q} \sum_{(u_1,u_2) \in R(q)} \log\left(1 + \exp\left(P\left(u_2 \mid q, i_{u_2}\right) - P\left(u_1 \mid q, i_{u_1}\right)\right)\right),$$

where $R(q)$ is a set of preferences for URLs from q's click-though data, i_u is the ranked position of u in the original search result.

5.4 Re-ranking Features

In our experiments, we adopt various features in different levels, including entity features, query features and URL features. These features are helpful in recognizing different situations and adjusting distribution into a better status. Note that these features are utilized as the feature set X in our model (See Sec. 5.2).

5.4.1 Entity Features

The entities of queries are treated as the search intents of queries. It is intuitive that entities with different properties might use different parameters. Therefore, we extract entity features from the information of entities.

Number of types. Entities might have different numbers of types in the knowledge base system. Because the consistent ranking model treats relevance distribution as aggregation over all types, different numbers of types might have different performances in the model.

Type Entropy. We then consider not only the number of types. Some types such as "person" and "location" are too general such that too many entities belong to such types. In other words, patterns of these types might not match URLs which are relevant to users. If most of an entity's types are general, the impact of the consistent ranking model should be lower. To verify that, we calculate the type entropy for an entity e as follows:

$$\sum_{t \in T(e)} -P(e \mid t) \cdot \log P(e \mid t).$$

Here we treat all entities have equal importance and compute entity distribution $P(e \mid t)$ by a uniform distribution.

Entity Frequency. Different entities might have different levels of popularity. More popular entities will lead to more queries about them shown in search engines. To improve overall performance, search engines will optimize for them more than other queries. For these queries, the impact of original results should be higher than the consistent model. On the other hand, less popular entities will gain less optimization from search engines. We hope that re-ranking can leverage information from entities of the same type and improve their performance. Here the frequency of an entity query is utilized as the entity frequency.

5.4.2 Query Features

A user types characters into a query and asks the search engine with his/her information needs. Such query strings might contain some information of user's search intents.

Query Length. Consider that each term in a query has its meaning in user's search intents, we assume that longer queries contain more information. Following this assumption, the original search engine might have better performance for longer queries.

Query Frequency. Similar to entity frequency, we assume that queries also have different levels of popularity. Generally, search engines optimize for head queries but tail queries. Accordingly, the parameters should be different for queries with different popularity.

5.4.3 URL Features

With many properties of URLs, URL features are extracted from each ranked URL respectively. Different URLs might also have different parameters while re-ranking. These URLs might contain more information.

Pattern Matching. In the consistent ranking model, URLs without any pattern-match will be ignored and receive zero relevance. In this case, original results should have a higher impact. We use a binary variable to represent whether a URL is matched by a pattern.

Original Position. Original position represents the relevance score given by the original ranker of search engine. A URL with a high position might be relevant enough and does not need an ensemble with the consistent ranking model.

Consistent Relevance. Similar to the feature of original position, relevance generated by the consistent ranking model might also affect parameter selection.

N-gram Similarity. URLs like personal and official sites might have higher relevance, although they might not be matched by any pattern. To verify this point, we calculate the n-gram similarity between the hosts of URLs and queries. It is highly possible that a URL is the official site of an entity if the host of URL is similar to the query. To cover official sites under public publishing services, we also calculate the n-gram similarity of whole URL strings to queries.

6. EXPERIMENTS

We conduct extensive experiments on large-scale datasets to evaluate the performance of our re-ranking model and to verify the effectiveness of the co-cluster hypothesis.

6.1 Datasets and Experimental Settings

Knowledge Base. We use Freebase [6] dumped in January 2014 as our knowledge base. Freebase classifies entities into types under various domains such as `people/person` and `book/author`. An entity might belong to multiple types in Freebase. Each type under a domain is treated as a query type in our work. We then remove very sparse types such that each type has at least five entities. Finally, there are 444 query types in our type set.

Datasets. Our experimental data is comprised of click-through data submitted to a commercial search engine in November 2013. After extracting entity queries, in total there are 56,466,534 queries for 847,682 distinct entities. By generalizing URL collections, there are 30,765 unique URL patterns utilized in our experiments. We use the queries submitted before 22 November, 2013 as the training data to train models. The remaining data is used as the testing data to evaluate performance. For the ground-truth, we adopt click-through data in search engine logs. We treat URLs with SAT-Clicks as the ground-truth. To evaluate performance in different situations, we divide the testing data into seven testing datasets with different conditions as shown in Table 1. The overall testing dataset consists of all queries for representing the general case. We also evaluate performance for queries with different levels of popularity. For

Table 1: The description of each testing dataset.

Dataset	Description
All	All queries in testing data.
Head	Queries with top 10% entity frequency.
Tail	Queries with bottom 10% entity frequency.
New	Queries which do not appear in training data.
Peo.	Queries with type `people/person`.
Loc.	Queries with type `location/location`.
Org.	Queries with type `organization/organization`.

Table 2: Performance of ranking consistency. All improvements of our method against the default ranking are significantly different at 99% level in a paired t-test.

Type	Default	Our Approach
Overall types	0.5671	0.5943 (+4.78%)
`people/person`	0.6410	0.6517 (+1.67%)
`location/location`	0.6327	0.6455 (+2.02%)
`organization/organization`	0.7533	0.7588 (+0.73%)
`celebrities/celebrity`	0.6306	0.6697 (+6.21%)
`music/album`	0.4589	0.4842 (+5.51%)
`book/book`	0.5367	0.5544 (+3.31%)

entities without appearance in the training data, we would like to verify if our model can still make improvements. Finally, we evaluate the performance of queries with different types. Note that we choose those three types because they are the types with the most entities.

6.2 Experimental Results

6.2.1 *Evaluation of Ranking Consistency*

Because our work is the first study on ranking consistency among same-type queries in web search, there is no standard metric to evaluate that. Here we propose a new metric based on Kendall's tau [19], which is a rank correlation coefficient, to evaluate ranking consistency. For a type t and a set of queries $Q(t)$ with the type t, we define the ranking consistency with ranking correlations of all query pairs as follows:

$$\frac{1}{\binom{|Q(t)|}{2}} \sum_{q_1 \in Q(t)} \sum_{q_2 \in Q(t) \setminus q_1} \boldsymbol{\tau}\left(r\left(q_1, t\right), r\left(q_2, t\right)\right),$$

which $r(q, t)$ denotes the ranking result of t's URL patterns with query q, and $\boldsymbol{\tau}(r_1, r_2)$ is the standard Kendall's tau rank correlation coefficient [19]. To evaluate consistency for all patterns, we give zero rank scores to pattern without appearance in search results while calculating Kendall's tau.

Table 2 shows ranking consistencies of the default ranking and our method for overall types and some sampled types. Generally, our method can significantly improve the ranking consistencies over the original ranking results. Then we attempt to analyze improvements for each type. Regarding the first two types, it is obvious that their improvements are less than the overall cases. This is because they are too general such that their queries have many dissimilar search intents. For example, both queries with the types `basketball/player` and `film/actor` are used to hold the type `people/person`. They might originally have much different ranking results. The type `organization/organization` has the smallest improvement. A reason might be that it has much higher consistency in the default ranking results. The other reason is that the useful patterns of the type are too few to make a significant improvement. For the last three types, their improvements are more than the above ones. We explain that these types are more specific to groups of queries. It means that the queries have more similar URLs for these types. Also, they have larger weights in type distribution to queries such that the consistent ranking model ranks patterns of these types more consistently.

Table 3: Performance of pattern-type relevance estimation. All improvements against the baseline are significantly different at 95% level in a paired t-test.

Measure	Frequency	Our Approach
NDCG@1	0.9607	0.9821 (+2.23%)
NDCG@2	0.7655	0.8145 (+5.87%)
NDCG@3	0.6748	0.7363 (+8.61%)
NDCG@4	0.6267	0.6800 (+8.07%)
NDCG@5	0.5857	0.6450 (+9.69%)

6.2.2 *Evaluation of Pattern-Type Relevance*

Because the key of the consistent ranking model is to estimate the pattern-type relevance, we first evaluate performance of such relevance. For each type, we collect the five relevant URL patterns with highest estimated pattern-type relevance to be evaluated. For the baseline, we adopt a frequency-based model ranking each URL pattern by its clicked counts in the search engine logs.

In order to evaluate performance, we hire two assessors to manually judge collected URL patterns. For each collected pattern for a type, two assessors annotate it carefully and assign a three-level relevance score as follows: relevant and important (Score 5), generally relevant (Score 1) and irrelevant (Score 0). Note that all collected patterns are annotated by these assessors. With annotated results, we evaluate the quality of collected patterns with the metric NDCG@k as follows:

$$NDCG@k = \frac{DCG@k}{IDCG@k}, \quad DCG@k = \sum_{i=1}^{k} \frac{2^{rel_i} - 1}{\log_2(i+1)},$$

where rel_i is the relevance score of the i-th pattern, $IDCG@k$ is the ideal $DCG@k$. Note that we calculate $IDCG@k$ with k highest relevance scores (Score 5). In the annotation process, the agreement on annotated results by two assessors is 80.76%, with 0.65 unweighted kappa coefficient [7]. The final $NDCG$ is calculated as the average of $NDCG$ by results annotated by two assessors.

Table 3 represents the performance of pattern-type relevance estimation. Our approach significantly outperforms the frequency-based baseline. It shows that our method can rank relevant pattern-matched URLs higher in the consistent ranking model. The baseline's lower performance is caused by the biases from popular entities such that irrelevant patterns are ranked higher. On the other hand, our model considers such a problem and aggregates preferences by entities to achieve a better performance.

Table 4: Performance of re-ranking models and default ranking. All improvements of our methods against the default ranking are significantly different at 99% level in a paired t-test.

		Default	Our Approach (single params)	Our Approach (multiple params)
All	MAP	0.7272	0.7454 (+2.49%)	**0.7571 (+4.12%)**
	MRR	0.7288	0.7469 (+2.49%)	**0.7589 (+4.13%)**
Head	MAP	0.7294	0.7491 (+2.70%)	**0.7611 (+4.34%)**
	MRR	0.7309	0.7505 (+2.68%)	**0.7627 (+4.35%)**
Tail	MAP	0.7116	0.7228 (+1.57%)	**0.7384 (+3.76%)**
	MRR	0.7138	0.7251 (+1.58%)	**0.7408 (+3.78%)**
New	MAP	0.7272	0.7453 (+2.49%)	**0.7572 (+3.83%)**
	MRR	0.7287	0.7468 (+2.48%)	**0.7589 (+3.83%)**
Peo.	MAP	0.7468	0.7756 (+3.86%)	**0.7834 (+4.89%)**
	MRR	0.7483	0.7772 (+3.86%)	**0.7851 (+4.92%)**
Loc.	MAP	0.7268	0.7465 (+2.72%)	**0.7573 (+4.20%)**
	MRR	0.7283	0.7481 (+2.71%)	**0.7588 (+4.19%)**
Org.	MAP	0.8422	0.8615 (+2.28%)	**0.8674 (+2.99%)**
	MRR	0.8432	0.8624 (+2.28%)	**0.8684 (+3.00%)**

6.2.3 *Evaluation of Re-ranking Models*

Table 4 shows the experimental results of our re-ranking models and the default ranking on different testing datasets. As our evaluation metrics, we adopt the *mean reciprocal rank* (MRR) and *mean average precision* (MAP).

For the results of default ranking, the testing dataset of head queries is better than the overall performance. The reason is that search engines focus on optimizing such queries. On the other hand, tail queries have lower performance. Regarding testing datasets of specific types, although location queries have similar performance to overall cases, people and organizations queries have higher performance. One reason is that people and organization might have their own official sites. To some extent, these official sites are the most relevant to such queries, especially to organization queries. Search engines might notice that and optimize search results. With the above observations and analyses, the performance of default ranking seems reasonable.

For all testing datasets, our methods have significant improvements over the default ranking with both single parameter and multiple parameters at 99% level in a paired t-test. Generally, the model using multiple parameters achieves better performance than using a single parameter. It shows that our proposed features are also valuable in helping the model optimize parameters. Similar to the default ranking, our methods perform better on head entities than tail entities. This is because our model also optimizes parameters by each query although we generate pattern relevance by types. However, there are still great improvements in the queries of tail entities. It demonstrates that our approach can help improve the relevance ranking performance of the low-frequency queries. For queries without appearance in training data, our methods have similar improvements to the overall cases. This is the advantage of our methods which can cover new queries with the same types as training entities. Regarding testing datasets with specific types, the location queries have similar performance to the overall cases. Recall that we conclude that better performance of people queries in default rankings is caused by their own official sites. It is interesting that their improvements are still larger than the overall cases. We explain that there are many sites with URL patterns for people entities such as `www.biography.com`. For some specific groups of people, there are also special sites with URL patterns such as `www.imdb.com` for celebrities and `espn.go.com` for athletes. Learning to rank such patterns well helps the models achieve strong improvements. The other point is that the improvements of organization queries are less than overall cases. Our explanation is that there are few structured websites for multiple organization, so there are also few useful URL patterns to improve the performance.

Based on the experimental results and above analyses, we verify the effectiveness of our approach and its assistance in improving the relevance rankings with the original search results.

6.3 User Surveys

In this section, we would like to understand whether users favor consistent ranking results produced by our consistent ranking model (Stage 1). In order to conduct user surveys, we sample five types as search intents and five pairs of queries for each type. The sampled types consist of two human types (`basketball_player` and `film/actor`), two item types (`film.film` and `music/album`) and one location type (`location/location`). For each pair of queries, we extract their search results from a commercial search engine and keep web pages matched by URL patterns appearing in both search results. With these queries and results, we conduct two user surveys via Amazon Mechanical Turk[3]. In the first survey, we want to understand whether users favor consistent results when they observe two search results at the same time. For each query pair, we re-rank two original results by the consistent ranking model, and then ask all participants to decide whether the consistent results are more likely to rank relevant web pages higher. To avoid the description biases, we ensure that workers do not know which results are re-ranked ones and randomly arrange two kinds of results. The workers decide which kind of rankings is more relevant by their own observation and personal opinion. In the second survey, we show results of only one query in a question. We want to know whether users still favor consistent results when they do not have results of the other query for reference. Finally, there are 25 questions in the first survey and 50 questions in the second one. For each survey, we ask ten Mechanical Turk workers to answer all questions.

Figure 3 shows the results of two user surveys. When users observe two queries and their search results at the same time, it is obvious that users tend to prefer consistent results. It shows that users realize the differences and consider the consistent results are more relevant. In the second survey, the percentage selecting the original results increases when these queries appear separately. The reason is that users cannot directly observe the consistency between two ranking results. However, there are still more questions with more users selecting the results of consistent rankings. It indicates that users really care about the consistency of ranking results whether the results of queries appear at the same time or not. In other words, if we improve the consistency of current search engines, the overall relevance ranking performance may also improve.

7. CONCLUSIONS AND FUTURE WORK

In this paper, we propose a new idea called ranking consistency in web search. The actual observations show that

[3]Amazon Mechanical Turk: https://www.mturk.com/

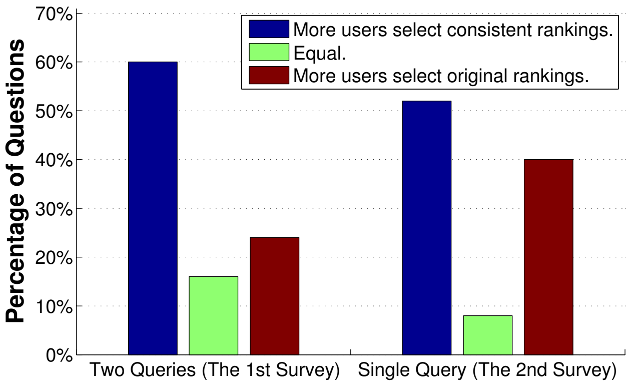


Figure 3: The results of two user surveys.

the search results of different queries with similar search intents should be consistent. However, conventional search engines optimize each query separately so that some queries may have inconsistent and less relevant search results. By leveraging knowledge base and click-through data, we propose a two-stage re-ranking model. The results of the extensive experiments demonstrate that our approach can indeed improve the ranking consistency of search results. In the meanwhile, the results show that our approach significantly improves the performance of both ranking consistency and relevance ranking. Such improvement is consistent across different datasets with different types of queries under different conditions. The reasons are as follows: (1) Our model considers queries with similar queries at the same time by leveraging a knowledge base to identify the types of queries; (2) Our approach precisely discovers the user preferences of URL patterns for each query type from query logs. Additionally, by leveraging the knowledge base and URL patterns , our model can effectively learn the ranking consistency for rare queries from the massive search logs of top queries with similar search intents. Moreover, two user surveys on Amazon Mechanical Turk also show that users are sensitive and prefer consistent search results. Overall, all experimental results demonstrate that the ranking consistency is an important factor in web search.

In future work, the performance may be much improved if it can be applied to modern state-of-the-art ranking models. Future work will consider applying the idea of the ranking consistency into state-of-the-art models for learning to rank.

8. REFERENCES

[1] P. N. Bennett, D. M. Chickering, and A. Mityagin. Learning consensus opinion: mining data from a labeling game. In *Proc. of WWW*, pages 121–130. ACM, 2009.

[2] P. N. Bennett, F. Radlinski, R. W. White, and E. Yilmaz. Inferring and using location metadata to personalize web search. In *Proc. of SIGIR*, pages 135–144. ACM, 2011.

[3] P. N. Bennett, K. Svore, and S. T. Dumais. Classification-enhanced ranking. In *Proc. of WWW*, pages 111–120. ACM, 2010.

[4] P. N. Bennett, R. W. White, W. Chu, S. T. Dumais, P. Bailey, F. Borisyuk, and X. Cui. Modeling the impact of short-and long-term behavior on search personalization. In *Proc. of SIGIR*, pages 185–194. ACM, 2012.

[5] J. Bian, T.-Y. Liu, T. Qin, and H. Zha. Ranking with query-dependent loss for web search. In *Proc. of WSDM*, pages 141–150. ACM, 2010.

[6] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proc. of SIGMOD*, pages 1247–1250. ACM, 2008.

[7] R. L. Brennan and D. J. Prediger. Coefficient kappa: Some uses, misuses, and alternatives. *Educational and psychological measurement*, 41(3):687–699, 1981.

[8] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender. Learning to rank using gradient descent. In *Proc. of ICML*, pages 89–96. ACM, 2005.

[9] B. Carterette and D. Petkova. Learning a ranking from pairwise preferences. In *Proc. of SIGIR*, pages 629–630. ACM, 2006.

[10] Y. Chen, X. Li, A. Dick, and R. Hill. Ranking consistency for image matching and object retrieval. *Pattern Recognition*, 47(3):1349–1360, 2014.

[11] M. Farah and D. Vanderpooten. An outranking approach for rank aggregation in information retrieval. In *Proc. of SIGIR*, pages 591–598. ACM, 2007.

[12] S. Fox, K. Karnawat, M. Mydland, S. Dumais, and T. White. Evaluating implicit measures to improve web search. *ACM Transactions on Information Systems (TOIS)*, 23(2):147–168, 2005.

[13] J. Gao, W. Yuan, X. Li, K. Deng, and J.-Y. Nie. Smoothing clickthrough data for web search ranking. In *Proc. of SIGIR*, pages 355–362. ACM, 2009.

[14] J. Guo, G. Xu, X. Cheng, and H. Li. Named entity recognition in query. In *Proc. of SIGIR*, pages 267–274. ACM, 2009.

[15] J. Hu, G. Wang, F. Lochovsky, J.-t. Sun, and Z. Chen. Understanding user's query intent with wikipedia. In *Proc. of WWW*, pages 471–480. ACM, 2009.

[16] J. Jiang, N. Yu, and C.-Y. Lin. Focus: learning to crawl web forums. In *Proc. of WWW*, pages 33–42. ACM, 2012.

[17] T. Joachims. Optimizing search engines using clickthrough data. In *Proc. of SIGKDD*, pages 133–142. ACM, 2002.

[18] T. Joachims, L. Granka, B. Pan, H. Hembrooke, F. Radlinski, and G. Gay. Evaluating the accuracy of implicit feedback from clicks and query reformulations in web search. *ACM Transactions on Information Systems (TOIS)*, 25(2):7, 2007.

[19] M. G. Kendall. A new measure of rank correlation. *Biometrika*, 30(1/2):81–93, 1938.

[20] K. Li, X. Cheng, Y. Guo, and K. Zhang. Crawling dynamic web pages in www forums. *Jisuanji Gongcheng/ Computer Engineering*, 33(6):80–82, 2007.

[21] X. Li, Y.-Y. Wang, D. Shen, and A. Acero. Learning with click graph for query intent classification. *ACM Transactions on Information Systems (TOIS)*, 28(3):12, 2010.

[22] J. Liu, Y.-I. Song, and C.-Y. Lin. Competition-based user expertise score estimation. In *Proc. of SIGIR*, pages 425–434. ACM, 2011.

[23] D. Metzler. Using gradient descent to optimize language modeling smoothing parameters. In *Proc. of SIGIR*, pages 687–688. ACM, 2007.

[24] T. M. Mitchell. *Machine Learning*. McGraw-Hill, Inc., 1997.

[25] T. H. Nguyen and B. K. Szymanski. Social ranking techniques for the web. In *Proc. of ASONAM*, pages 49–55. ACM, 2013.

[26] F. Radlinski and T. Joachims. Query chains: learning to rank from implicit feedback. In *Proc. of SIGKDD*, pages 239–248. ACM, 2005.

[27] Y.-I. Song, J.-T. Lee, and H.-C. Rim. Word or phrase?: learning which unit to stress for information retrieval. In *Proc. of ACL*, pages 1048–1056. ACL, 2009.

[28] H. Wang, X. He, M.-W. Chang, Y. Song, R. W. White, and W. Chu. Personalized ranking model adaptation for web search. In *Proc. of SIGIR*, pages 323–332. ACM, 2013.

[29] K. Wang, T. Walker, and Z. Zheng. Pskip: estimating relevance ranking quality from web search clickthrough data. In *Proc. of SIGKDD*, pages 1355–1364. ACM, 2009.

[30] W. Wu, H. Li, H. Wang, and K. Q. Zhu. Probase: A probabilistic taxonomy for text understanding. In *Proc. of SIGMOD*, pages 481–492. ACM, 2012.

[31] X. Yin, W. Tan, X. Li, and Y.-C. Tu. Automatic extraction of clickable structured web contents for name entity queries. In *Proc. of WWW*, pages 991–1000, 2010.