Mining Search Engine Clickthrough Log for Matching *N*-gram Features

Huihsin Tseng, Longbin Chen, Fan Li, Ziming Zhuang,
Lei Duan, Belle Tseng
Yahoo! Inc., Santa Clara, CA 95054
{huihui,longbin,fanli,ziming,leiduan,belle}@yahoo-inc.com

Abstract

User clicks on a URL in response to a query are extremely useful predictors of the URL's relevance to that query. Exact match click features tend to suffer from severe data sparsity issues in web ranking. Such sparsity is particularly pronounced for new URLs or long queries where each distinct query-url pair will rarely occur. To remedy this, we present a set of straightforward yet informative query-url *n*-gram features that allows for generalization of limited user click data to large amounts of unseen query-url pairs. The method is motivated by techniques leveraged in the NLP community for dealing with unseen words. We find that there are interesting regularities across queries and their preferred destination URLs; for example, queries containing "form" tend to lead to clicks on URLs containing "pdf". We evaluate our set of new query-url features on a web search ranking task and obtain improvements that are statistically significant at a *p*-value < 0.0001 level over a strong baseline with exact match clickthrough features.

1 Introduction

Clickthrough logs record user click behaviors, which are a critical source for improving search relevance (Bilenko and White, 2008; Radlinski et al., 2007; Agichtein and Zheng, 2006; Lu et al. 2006). Previous work (Agichtein et al., 2006) demonstrated that clickthrough features (e.g., IsNextClicked and IsPreviousClicked) can lead to substantial improvements in relevance. Such features summarize query-specific user interactions on a search engine. One commonly used clickthrough feature is generated based on the following observation: if a URL receives a large number of first and last clicks across many user sessions, then it indicates that this URL might be a strongly preferred destination of a query. For example, when a user searches for "yahoo", they tend to only click on the URL www.yahoo.com rather than other alternatives. This results in www.yahoo.com being the first and last clicked URL for the query. We refer to such behavior as being navigational clicks (**NavClicks**). Features that use exact query and URL string matches (e.g., NavClick, IsNextClicked and IsPreviousClicked) are referred to as exact match features (**ExactM**) for the remainder of this paper.

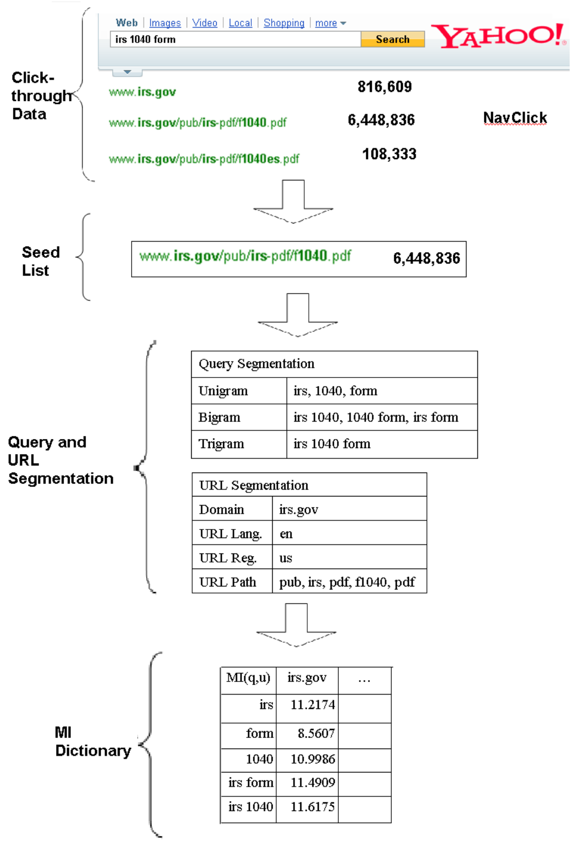
The coverage of ExactM features is sparse, especially for long queries and new URLs. Many long queries are either unique or very low frequency. Hence, the improvements from ExactM features are limited to the more popular queries. In addition, ExactM features tend to be weighted heavily in the ranking of results when they are available. This introduces a bias where the ranking models tend to strongly favor older URLs over new URLs even when the latter otherwise appear to be more relevant.

By inspecting the clickthrough logs, we observed that unseen query-url pairs are often composed of informative previously observed subsequences. Specifically, we saw that query *n*-grams can be correlated with sequences of URL *n*-grams. For example, we find that there are interesting regularities across queries and URLs, such as queries containing "form" tending to lead to clicks on URLs containing "pdf". This strongly motivates the adoption of an approach similar to the Natural Language Processing (**NLP**) technique of using *n*-grams to deal with unseen words. For example, part-of-speech tagging (Brants, 2000) and parsing (Klein and Manning, 2003) both require dealing with unknown words. By using *n*-gram substrings, novel items can be dealt with using any informative substrings they contain that were actually observed in the training data.

The remainder of the paper is organized as follows. In Section 2, we introduce our overall methodology. Section 2.1 presents a data mining method for building a query-url *n*-gram dictionary, Section 2.2 describes the new ranking features in detail. In section 3, we present our experimental results. Section 4 discusses related work, and Section 5 summarizes the contribution of this work.

2 Methodology

This section describes the detailed methodology used in generating the query-url *n*-gram features. Our features require association scores to be previously calculated, and, hence, we first introduce a data mining approach that is used to build an association dictionary in Section 2.1. Then, we present the procedure used to generate the query-url *n*-gram features that use the dictionary in Section 2.2.


Web | Images | Video | Local | Shopping | more
irs 1040 form
Search
YAHOO!
Click-through Data
www.irs.gov 816,609
www.irs.gov/pub/irs-pdf/f1040.pdf 6,448,836
NavClick
www.irs.gov/pub/irs-pdf/f1040es.pdf 108,333
Seed List
www.irs.gov/pub/irs-pdf/f1040.pdf 6,448,836
Query and URL Segmentation
MI Dictionary

Query Segmentation	
Unigram	irs, 1040, form
Bigram	irs 1040, 1040 form, irs form
Trigram	irs 1040 form

URL Segmentation	
Domain	irs.gov
URL Lang.	en
URL Reg.	us
URL Path	pub, irs, pdf, f1040, pdf

MI(q,u)	irs.gov	…
irs	11.2174	
form	8.5607	
1040	10.9986	
irs form	11.4909	
irs 1040	11.6175	

Figure 1: Steps to build a query-url *n*-gram dictionary

2.1 Data Mining on a Query-URL *N*-gram Dictionary

The steps involved in building the dictionary are shown in Figure 1. We first collect seed query-url pairs from clickthrough data based on NavClicks. The queries and URLs from the collected pairs are tokenized and converted into a collection of paired query-url *n*-grams. For each pair, we calculate the mutual information of the query *n*-gram and its corresponding URL *n*-gram. For our experiment, we collect a total of more than 15M seed pairs and 0.5B query-url *n*-gram pairs using six months of query log data. The details are described in the following sections.

2.1.1 Seed List

We identify the seed list based on characteristic user click behavior. Given a query, we select the URL with the most NavClicks as compared to other URLs returned. During data collection, the rank positions of the top 5 URLs were shuffled to avoid the position bias. We aggregate NavClicks for a URL occurring in these positions in order to both obtain more click data and to avoid the position bias issue discussed in Dupret and Piwowarski (2008) and Craswell et al. (2008).

For example, in Figure 1, the numbers of NavClicks for the top three URLs are shown. The URL www.irs.gov/pub/irs-pdf/f1040.pdf receives the largest number of NavClicks, and, therefore, it is used to create the query-url pair:

[irs 1040 form, www.irs.gov/pub/irs-pdf/f1040.pdf]

2.1.2 Query and URL Segmentation

We segment the seed pairs to *n*-gram pairs in order to increase the coverage beyond that of ExactM click features. Within NLP, *n*-grams are typically extracted such that words that are adjacent in the original sequence are also adjacent in the extracted *n*-grams. Furthermore, we attempt to achieve additional generalization by using skip *n*-grams (Lin and Och, 2004). This means we not only extract *n*-grams for adjacent terms but also for sequences that leave out intermediate terms. This is motivated by the observation that the semantics of user queries is often preserved even when some intermediate terms are removed. The details of the segmentation methods are described below.

2.1.2.1 Query Segmentation

Prior to query segmentation, we normalize raw queries by replacing punctuations with spaces. Queries are then segmented into a sequence of space delimited tokens. From these, we extract all possible query *n*-grams and skip *n*-grams for *n* smaller than or equal to three (i.e., all unigrams, bigrams, and trigrams). For example, given the sequence "irs 1040 form" the adjacent bigrams would be "irs 1040" and "1040 form". With skip *n*-grams we also extract "irs form" as shown in Table 1. We do not use *n*-grams longer than 3 in order to avoid problems with overfitting. We will refer to this segmentation method as **Affix Segmentation**.

Table 1: An Example of Affix Segmentation

N-gram	Affix Segmentation
Unigram	irs, 1040, form
Bigram	irs 1040, 1040 form, irs form
Trigram	irs 1040 form

2.1.2.2 URL Segmentation

As shown in Table 2, after the queries are segmented, URLs are categorized into four groups: domain, URL language, URL region and URL path. In general, a URL is delimited by punctuation characters such as "?", "."," "/", and "=".

Table 2: An Example of URL Segmentation

URL Groups	Example
Domain	irs.gov
URL language	en
URL region	us
URL path	pub, irs, pdf, f1040, pdf

The domain group includes one domain token, for example, irs.gov. Although domains could be divided into multiple *n*-grams, we treat them as a single unit, with the exception of encoded language and region information.

The language and region groups are based on the language or region part of the URL *n*-grams such as the suffixes ".en" and ".de". The language and region of a URL *n*-gram are identified by a table look-up method. The table is created based on the information available at en.wikipedia.org/wiki/List_of_ISO_639-1_codes and en.wikipedia.org/wiki/ISO_3166. When there is no clear language or region URL *n*-gram, we use English (en) as the default language and United States (us) as the default region.

2.1.3 Calculation of Mutual Information

After query and URL *n*-grams are extracted, we calculate mutual information (Gale and Church, 1991) to determine the degree of association between the *n*-grams. The definition of query-url *n*-gram mutual information (MI) is given in Equation 1.

$$\mathrm{MI}(q,u) = \log_2 \frac{\mathrm{Freq}(q,u)}{\mathrm{Freq}(q)\,\mathrm{Freq}(u)} \quad (1)$$

Here *q* corresponds to a query *n*-gram and *u* corresponds to a URL *n*-gram. Freq (*q*) is the count of q in the seed list normalized by the total number of *q*. Freq (*u*) is the count of u normalized by the total number of *u*. Freq (*q*, *u*) is the count of *q* and *u* that co-occurred in a full query-url pair normalized by the total number of *q* and *u*. A pair will be assigned to a MI score of zero if the items occur together no more than expected by chance, under the assumption that the two items are statistically independent. When a pair occurs more than is expected by chance, the MI score is positive. On the other hand, if a pair occurs together less than is expected by chance, the mutual information score is negative. In order to increase the confidence of the MI scores, we remove all *n*-grams with less than 3 occurrences in the seed list, and assign a zero MI score for any pairs involving these *n*-grams. No smoothing is applied.

This scoring scheme fits well with the association properties we would like to have for our query-url *n*-gram click features. If a query *n*-gram cues for a certain URL through one of its *n*-grams, the feature will take on a positive value. Similarly, if a query *n*-gram cues against a certain URL, the feature will take on a negative value.

2.1.4 Analysis of Query-URL *N*-gram Association

By examining our dictionary, we observed a number of pairs that are interesting from a relevance ranking perspective. To illustrate, we present four examples of *n*-gram pairs and intuitively explore the nature of the *n*-gram associations in the dictionary.

Table 3: Examples of MI Scores

Query *n*-gram	URL *n*-gram	MI score
"iphone"	apple.com	8.7713
"iphone"	amazon.com	-0.1555
"iphone plan"	att.com	11.5388
"iphone plan"	apple.com	8.9676

First, let's examine the association between query *n*-grams and URL *n*-grams for the queries "iphone" and "iphone plan". Notice that the query unigram "iphone" is strongly associated with apple.com, but negatively associated with amazon.com. This can be explained by the fact that "iphone" as a product is not only developed by Apple but also strongly associated with the Apple brand. In contrast, while Amazon.com sells iphones, it also sells a large variety of other products, thus is not regarded as a very authoritative source of information about the "iphone". However, by adding additional context, the most preferred URL according to MI can change. The two examples in the bottom of Table 3 illustrate the URL preferences for the query bigram "iphone plan". While apple.com is still a strongly preferred destination, there is a much stronger preference for att.com. This preference follows since apple.com has more product information on the "iphone" while the information provided by att.com will be more targeted at visitors who want to explore what rate plans are available.

Second, Table 4 shows the association between "kimo", ".tw" and ".us". "Kimo" was a Taiwanese start-up acquired by Yahoo!. The mutual information scores accurately reflect the association between the query *n*-gram and region ids.

Table 4: Example of MI Scores

Query *n*-gram	URL *n*-gram	MI score
"kimo"	tw (taiwan)	12.8303
"kimo"	us (united states)	0.7209

Third, Table 5 shows the association between "kanji", and URLs with Language identification of "Japanese", "Chinese" and "English". "Kanji" means "Chinese" in Japanese. Since queries containing "Kanji" are typically from users interested in Japanese sites, the mutual information shows higher correlation with Japanese than with English or Chinese.

Table 5: Example of MI Scores

Query *n*-gram	URL *n*-gram	MI score
"kanji"	ja (japanese)	11.3862
"kanji"	zh (chinese)	6.2567
"kanji"	en (english)	4.2110

Table 6: Example of MI Score

Query *n*-gram	URL *n*-gram	MI score
"form"	pdf	4.9067
"form"	htm	1.0916
"video"	watch	5.7192
"video"	htm	-1.9079

Fourth, Table 6 shows the association between two query *n*-grams, "form" and "video", that at first glance may not actually look very informative for URL path selection. However, notice that the unigram "form" has a strong preference for pdf documents over more standard web pages with an html extension. Similarly, queries that include "video" convey a preference for URLs containing "watch", a characteristic URL *n*-gram for many video sharing websites.

It is reasonable to anticipate that incorporating such associations into a search engine's ranking function should help improve both search quality and user experience. Take the example where, there are two high ranking competing URLs for the query "irs 1040 form". Let's also assume both documents contain the same query relevant keywords, but one is an introduction of the "irs 1040 form" as an htm webpage and the other one is the real filing form given as a pdf document. Since in our dictionary, "form" is more associated with pdf than htm, we predict that most users would prefer the real pdf form directly, so it should be placed first in the list of query results. While click data for the exact query-url pairs confirms this preference, it is reassuring that we could identify it without needing to rely on seeing the specific query string before. As described in detail below, and motivated by this analysis, we designed our query-url click features based on the contents of the *n*-gram MI dictionary.

2.2 Query-URL *N*-gram Features

For our feature set, we explored the use of different query segmentation approaches (concept and affix segmentation) in order to increase the diversity of *n*-grams. In the following section, we use an unseen query "irs 1040 forms" and contrast it with the known query "irs 1040 form" from the last section.

2.2.1 Concept Segmentation Features

Query concept segmentation is a weighted query segmentation approach. Each query is analytically interpreted as being a main concept and a sub concept. We search for the unique segmentation of the query that maximizes its cumulative mutual information score with the URL *n*-grams. Main concepts and sub concepts are *n*-grams from the query that have the strongest association with URL *n*-grams and thus assist in identifying relevant landing URL *n*-grams when the whole query or the whole URL has not been seen.

Algorithm 1: Concept Segmentation

```
for U = domain, URL language, URL region, URL path do
    for $j$ = 0... $n-1$ do
        M $\Leftarrow$ $W_{0...j}$
        S $\Leftarrow$ $W_{j+1...n}$
        for $k$ = 0... $m$ do
            curr_mi_M $\Leftarrow$ arg $\max_{k=1...m}$ MI (M, $U_k$)
            curr_mi_S $\Leftarrow$ arg $\max_{k=1...m}$ MI (S, $U_k$)
            if curr_mi_M + curr_mi_S > curr_best then
                curr_best = curr_mi_M + curr_mi_S
                mi_M $\Leftarrow$ curr_mi_M
                mi_S $\Leftarrow$ curr_mi_S
            end if
        end for
        adding mi_M as a feature
        adding mi_S as a feature
    end for
end for
```

Pseudo-code for generating query-url *n*-gram features based on the concept segmentation is given in Algorithm 1. Each query (Q) is composed of a number of words, w_1, w_2, w_3…,w_n. Each URL is segmented and categorized to four groups: domain, URL language, URL region and URL path. Each URL group has *m* number of URL *n*-grams. M is the main concept of Q and S is the sub concept of Q.

One potential drawback of such concept segmentation is data sparsity. When we look for the maximum of cumulative mutual information, we may obtain main concepts with very high mutual information and sub concepts which do not exist in the dictionary. In order to address this problem, we implement a second query segmentation method, affix segmentation, that is discussed in section 2.2.2.

Table 7 shows eight concept segmented features. "Coverage" is the percentage of query-url pairs that have valid feature values. Some of the samples do not have values because no clicks for the pairs were seen in the sample of data used to build the dictionary. When a pair does not have a value, the default value of zero is assigned. This default value is based on the assumption that unless we have evidence otherwise, we assume all query-url *n*-grams are statistically independent and thus provide no preference signal.

Table 7: Eight Features Generated based on Concept Segmentation.

Feature	Query *N*-gram	URL *N*-gram	Coverage (%)
MainDS	M	domain	54.09
SubDS	S	domain	30.46
MainLang	M	lang.	94.41
SubLang	S	lang.	72.40
MainReg	M	reg.	90.34
SubReg	S	reg.	68.19
MainPath	M	path	64.96
SubPath	S	path	58.76

Query-URL Domain Features are defined as the mutual information of a query *n*-gram and the domain level URL. There are two features in this category, one for the query main concept and one for the sub concept. They help to identify the user preferred host given a query.

Table 8: Example of Selecting Query Segmentation

MI(q,u)	irs.gov	
"irs"		11.2174
"1040"	11.6175	11.5550
"forms"	7.5049	
Cumulative MI	19.1224	22.7724
	Seg. 1	Seg.2

To illustrate the concept segmentation features, let's examine the query, "irs 1040 forms" in the context of the domain irs.gov. The query "irs 1040 forms" can be segmented either as "irs 1040" and "forms" or as "irs" and "1040 forms". As shown in Table 8, taking the cumulative maximum, the second segmentation scores higher than the first one. Therefore, the "irs" and "1040 forms" segmentation is preferred. The feature value for the main concept is 11.5550, and the sub concept is then assigned to be 11.2174.

Query-URL Language and Region Features are the mutual information of a query *n*-gram and URL language/region. They are used for providing language and region information.

Query-URL Path Features are the mutual information of a query *n*-gram and a URL path *n*-gram. While there are typically many URL path *n*-grams, only one URL path *n*-gram is selected to be paired with each query *n*-gram. The selected *n*-gram is the one that achieves the highest cumulative maximum MI score. They are used for providing association between query *n*-grams and url *n*-grams such as “forms” and “pdf”.

2.2.2 Affix Segmentation Features

As previously mentioned, affix segmentation addresses sparsity issues associated with concept segmentation. Here, we introduce the features generated based on affix segmentation. Pseudo-code for generating the features is given in Algorithm 2. Two query unigrams (w_0 and w_n) and one bigram (w_0w_n) is used. Each URL is segmented and categorized to four groups: domain, URL language, URL region and URL path. Each URL group has *m* number of URL *n*-grams.

This approach is complementary to the concept segmentation for long queries. The affix *n*-grams are in smaller unit, and therefore, are less sparse. In addition, the skip bigrams allow for generalizations using non-adjacent terms. Table 9 shows the coverage of the twelve affix features.

Algorithm 2: Affix Segmentation

```
for U = domain, URL language, URL region, URL path do
  for q = $w_0$, $w_n$, $w_0w_n$ do
    for $k$ = 0... $m$ do
      curr_mi_q ⇐ arg max$_{k=1...m}$ MI (q, $U_k$)
      if curr_mi_q > curr_best then
        curr_best = curr_mi_q
      end if
    end for
    adding curr_mi_q as a feature
  end for
end for
```

Table 9: Twelve Features Generated based on Affix Segmentation

Feature	Query *N*-gram	URL *N*-gram	Coverage (%)
PreDS	w_0	domain	48.09
SufDS	w_n	domain	47.72
PresufDS	w_0w_n	domain	23.57
PreLang	w_0	lang.	55.58
SufLang	w_n	lang.	58.22
PresufLang	w_0w_n	lang.	24.91
PreReg	w_0	reg.	93.82
SufReg	w_n	reg.	93.59
PresufReg	w_0w_n	reg.	69.29
PrePath	w_0	path	98.15
SufPath	w_n	path	97.80
PresufPath	w_0w_n	path	75.81

Query-url domain affix features has three features: MI(w_0, domain), MI(w_n, domain), and MI(w_0w_n, domain). In the example of “irs 1040 forms” and “irs.gov”, the features are MI(irs, irs.gov), MI(forms, irs.gov), and MI(irs forms, irs.gov).

Query-url language and region affix features has three features respectively: MI(w_0, language), MI(w_n, language), MI(w_0w_n, language) MI(w_0, region), MI(w_n, region), and MI(w_0w_n, region). In the example of “irs 1040 forms”, “en” and “us”, the features are MI (irs, en), MI (forms, en), MI (irs forms, en), MI (irs, us), MI (forms, us), and MI (irs forms,us).

Query-url path affix features has three features: MI(w_0, path), MI(w_n, path), and MI(w_0w_n, path). In the example of “irs 1040 forms” and “www.irs.gov/pub/irs-pdf/f1040.pdf”, there are four URL path *n*-grams, “pub”, “irs”, “pdf”, and “f1040”. The URL path *n*-gram, irs, gets maximum MI score. Therefore, the query-url path affix features are MI (irs, irs), MI (forms, irs), and MI (irs forms, irs).

We demonstrated the procedure to generate 20 query-url *n*-gram features, and in Section 3, we will present their effectiveness in relevance ranking.

3 Experiment

We evaluate the performance of query-url *n*-grams features (8 concept and 12 affix features) on a ranking application and analyze the results from several different perspectives.

3.1 Datasets

For all experiments, our training and test data are query-url pairs annotated with human judgments. In our data, we use five grades to evaluate relevance of a query and URL pair.

The data includes 94K queries for training and 3.4K queries for evaluation, and each query is associated with the top ranked URLs returned from a search engine. Totally, there are 916K query-url pairs for training and 42K pairs for testing. The queries are general and uniformly and randomly sampled with replacement, resulting in more frequent queries also appearing more frequently in our training and test sets.

3.2 Ranking Algorithm

GBRank is a supervised learning algorithm that uses boosted decision trees and incorporates the pair-wise information from the training data (Zheng et al, 2007). It is able to deal with a large amount of training data with hundreds of features. We use an internal C++ implementation of GBRank.

3.3 Evaluation Metric

We use Discounted Cumulative Gain (Järvelin and Kekäläinen, 2002) to evaluate our ranking accuracy. Discounted Cumulative Gain (DCG) has been widely used in evaluating the quality of search engine rankings and is defined as:

$$DCG_k = \sum_{i=1}^{k} \frac{G_i}{\log_2(i+1)} \quad (2)$$

G_i represents the editorial judgment of the i-th document. In this paper, we only report **normalized DCG$_5$**, which is an absolute DCG$_5$ normalized by a baseline, and **relative DCG$_5$ improvement**, which is an improvement normalized by the baseline. Note normalized DCG$_5$ is different than NDCG (Normalized Discounted Cumulative Gain defined in Järvelin and Kekäläinen, 2002). We use Wilcoxon signed test (Wilcoxon, 1945) to evaluate the significance for model comparison.

3.4 Feature Sets

Five feature sets are used in our experiments. Details are listed in Table 10.

Table 10: Five Feature Sets

Tag	Description
Base Feature Set	Core Feature Set and ExactM click features
Q-U *N*-gram Feature Set (I)	Base Feature Set and Q-U *N*-gram features
Core Feature Set	query-based, document-based, query-document based features
NavClick Feature Set	Core Feature Set and NavClick
Q-U *N*-gram Feature Set (II)	Core Feature Set and Q-U *N*-gram features

Base Feature Set is a strong baseline feature set from a state-of-the-art commercial search engine. This set includes NavClick features, and other internal ExactM click features. It is used for evaluating Query-URL *N*-gram Feature Set (I) in order to know whether query-url *n*-gram features can achieve gains when stacked on top of ExactM features.

Core Feature Set is a weaker variant of the baseline system that excludes ExactM click features. This system is used for evaluating NavClick Feature Set and Query-URL *N*-gram Feature Set (II) independently in order to study and contrast the effected queries.

3.5 Experimental Results

We compare the query-URL *N*-gram feature set (I) with the base feature set in Section 3.5.1, and contrast the NavClick features and the query-URL *N*-gram features (II) using the Core Feature Set in Section 3.5.2.

3.5.1 Query-URL *N*-gram Feature Set (I) versus Base Feature Set

As shown in Figure 2, Query-URL *N*-gram Feature Set (I) outperforms Base Feature Set. The additional 20 query-url *n*-gram features achieve statistically significant gains at a p-value < 0.0001 level, suggesting that they are complimentary to ExactM click features. Even though the query-url *n*-gram features are generated from the same data as the ExactM features, the gain is additive and stackable. The DCG$_5$ impact is 0.53% relative improvement when running GBRank using 2500 trees. Every data point is normalized by the DCG$_5$ of the baseline feature set using 2500 trees. This is represented in the graph as the rightmost point of Base Feature Set curve.

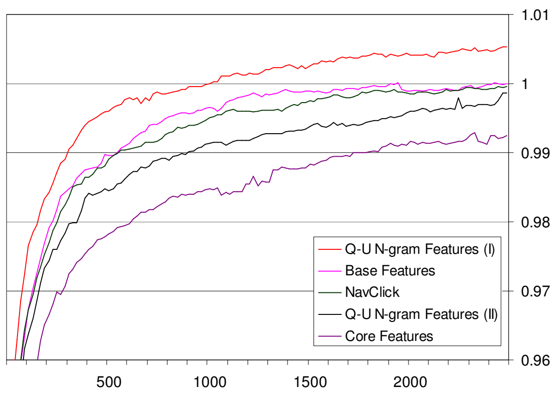


Figure 2: Comparison of the five feature sets on the normalized DCG$_5$ (Y-axis) against number of trees (X-axis).

3.5.2 NavClick and Query-URL *N*-gram Feature Set (II) versus Core Feature Set

We compare NavClick Feature Set and Query-URL *N*-gram Feature Set (II) in the context of Core Feature Set, in order to evaluate the two independently. As shown in Figure 2, both NavClick and Query-URL *N*-gram Feature Set (II) outperform Core Feature Set. It is not surprising that NavClick also outperforms Query-URL *N*-gram Feature Set (II) since the *n*-gram features are backoff of NavClick. However, their gains are competitive suggesting the query-url *n*-gram features are very good relevance indicators. The impact of NavClick and Query-URL *N*-gram Feature Set (II) is 0.72% and 0.62% relative DCG_5 improvement at Tree 2500 respectively.

3.5.3 Feature Importance

Using the GBRank model, features are evaluated and sequentially selected to build the boosted decision trees. The split of each node increases the DCG during training. We evaluate a feature's importance by aggregating the DCG impact of the feature over all trees (Zheng et al., 2007). Here, the feature importance is rescaled so that the feature with largest DCG impact is assigned a normalized score of 1. Figure 3 illustrates the relative influence of each of query-url *n*-gram feature. Of these, *n*-gram features associated with a domain name (i.e., MainDS) rank highest.

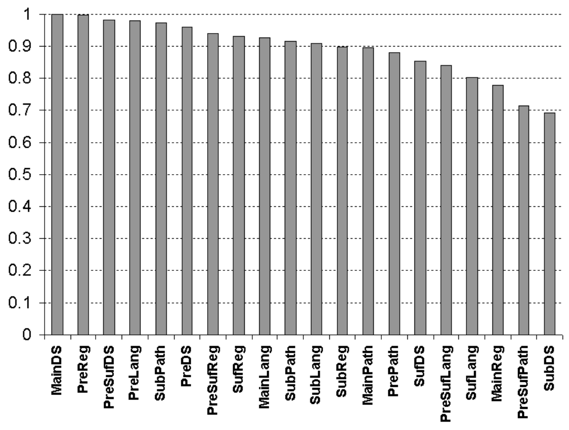


Figure 3: Feature importance of query-url *n*-gram features. The importance (Y axis) is normalized so that the most important feature (MainDS)'s importance is 1.

3.6 Analysis

We access system performance with respect to both query length and frequency using the two click features sets in combination with the Core Feature Set in order to gain insight into the effected queries.

3.6.1 Query Length

As shown in Table 11, NavClick (NavClick Feature Set) best improves relevance for two word queries. In contrast, Query-url *n*-gram features in isolation (Query-URL *N*-gram Feature II) are able to show sizable improvements on longer queries, while slightly degrading performance on short 1-word queries. Using both feature sets together (Query-URL *N*-gram Feature I) results in improvement for queries of all lengths.

These results suggest that the strong signal being provided by NavClick for short queries helps to compensate for any additional noisy introduced by the *n*-gram features, while allowing the *n*-gram features to handle longer queries that are less well covered by NavClick. These longer queries are exactly the type of queries our query-url *n*-gram features were designed to help with.

Table 11: Relative DCG_5 Improvement of NavClick, Query-URL *N*-gram (II), and Query-URL *N*-gram Features (I) vs Core Feature Set

Length	NavClick vs Core (%)	QU *N*-gram (II) vs Core (%)	QU *N*-gram (I) vs Core (%)
1 word	0.03	-0.04	0.62
2 words	1.04	1.06	1.58
3 words	1.00	1.44	2.12
4+ words	0.4	0.68	1.01

3.6.2 Query Frequency

We found that query-url *n*-gram features improve tail queries. Head queries are considered as top two million frequent queries in our traffic and tail queries include anything outside of that range.

Table 12: Relative DCG_5 Improvement of NavClick, Query-URL *N*-gram Features (II) and Query-URL *N*-gram Features (I) vs Core Feature Set

	NavClick vs Core (%)	QU *N*-gram (II) vs Core (%)	QU *N*-gram (I) vs Core (%)
Head	0.91	-0.15	1.11
Tail	0.59	1.11	1.40

As shown in Table 12, query-url *n*-gram features (Query-URL Feature Set II) differ from NavClick (NavClick Feature Set) in that they get more gain from tail queries. Together, they (Query-URL Feature Set I) improve both head and tail queries.

3.7 Case Study

Below we examine queries from the test set and analyze the effects of Query-URL *N*-gram Feature Set (II) versus Core Feature Set.

3.7.1 Positive Cases

1) Animal shelter in va: this query targets a specific geographic location. Using the baseline feature set, the root url wvanimalshelter.org is incorrectly ranked higher than www.netpets.com/cats/catresc/virginia.htm. Without any additionally ranking information, general URLs (root) tend to be ranked more highly than more specific URLs (path), as the root pages tend to be more popular. However, our new features express a preference between "va" and "virginia", and this correctly flips the ranking order.
2) Myspace profile generator: www. myspacgens.com/handler.php?gen=profile was incorrectly ranked higher than www.profilemods.com/myspace-generators. Our new features convey a high user preference association between "profile generator" and the domain profilemods.com, which helps to correctly swap the order.

3.7.2 Negative Cases

We determined that negative cases where the baseline feature set outperforms the new features are typically one word navigational queries such as "craigslist". However, after we combine the query-url *n*-gram features with NavClick, one word navigational queries are ranked correctly.

4 Related Work

Our work is mainly related to Gao et al. (2009) and Bilenko and White (2008). Gao et al. (2009) addressed the sparsity issue by propagating click information among similar queries in the same cluster. Their idea is based on an observation that similar queries go to similar pages. When two queries have similar clicked URLs, it is likely that they share clicked URLs. In contrast, our idea is to utilize NLP techniques to break down long, infrequent queries into shorter, frequent queries. The two approaches can be mutually beneficial. Bilenko and White (2008) expanded click data with a search engine by using post-search user experience collected from toolbars. Toolbars keep track of users' click behavior both when they are using the search engine directly and beyond. Their relevance features are built based on whole session clicks extracted from the toolbar. In contrast, our *n*-gram features are built on search engine clicks directly. We should be able to expand our method to integrate the post-search clicks with toolbar data.

Other related work can be found in the domain of query rewriting. Our *n*-gram dictionary was originally designed for query rewriting. Query rewriting (Xu and Croft, 1996; Salton and Voorhees, 1984) reformulates a query to its synonyms or related terms automatically. However, the coverage of query rewriting is normally small, because an inappropriate rewrite can cause significant decrease in precision. In contrast, our approach can cover a larger number of queries without decreasing precision, because it does not need to make a binary decision whether a query should be reformulated. The association scores between queries and rewrites are used as ranking features which are trained discriminatively toward search quality.

5 Conclusion

In this paper, we presented a set of straightforward yet informative query-url *n*-gram features. They allow for generalization of limited user click data to large amounts of unseen query-url pairs. Our experiments showed such features gave significant improvement over models without using the features. In addition, we mined an interesting dictionary which contains informative, but not necessarily obvious, query-url synonym pairs such as "form" and "pdf". We are currently extending our work to a variety of exact match features and different sources of clickthrough logs.

Acknowledgement

Thanks to the anonymous reviewers for detailed suggestion and our colleagues: Jon Degenhardt and Narayanan Sadagopan for assistance on generating clickthrough data, Jiang Chen for developing the decision tree package, Xiangyu Jin for a discussion on map/reduce, Benoît Dumoulin, Fuchun Peng, Yumao Lu, and Xing Wei for productizing the work, and Rosie Jones, Su-lin Wu, Bo Long, Xin Li and Ruiqiang Zhang for comments on an earlier draft.

References

Agichtein, E., E. Brill, and S. Dumais. 2006. Improving web search ranking by incorporating user behavior information. In Proceedings of the ACM SIGIR 29.

Agichtein, Eugene, Zijian Zheng. 2006. Identifying "best bet" web search results by mining past user behavior. In Proceedings of KDD.

Bilenko, Mikhail and Ryen W. White. 2008. Mining the search trails of surfing crowds: identifying relevant websites from user activity. In Proceedings of WWW.

Brants, T. 2000. Tnt: a statistical part-ofspeech tagger. In Proceedings of ANLP 6.

Craswell, Nick and Martin Szummer. 2007. Random walks on the click graph. In Proceedings of SIGIR.

Craswell, Nick, Onno Zoeter, Michael Taylor, Bill Ramsey. 2008. An experimental comparison of click position-bias models in WSDM.

Dupret, Georges, Benjamin Piwowarski. 2008. A user browsing model to predict search engine click data from past observations. In Proceedings of SIGIR 31.

Gale, William A. and Kenneth W. Church. 1991. Identifying word correspondence in parallel texts. In Proceedings of HLT 91.

Gao, Jianfeng, Wei Yuan, Xiao Li, Kefeng Deng, and Jian-Yun Nie. 2009. Smoothing Clickthrough Data for Web Search Ranking. In Proceedings of SIGIR 32.

Järvelin, K. and J. Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques, Journal ACM Transactions on Information Systems, 20: 422-446.

Klein, D. and C. Manning. 2003. Accurate unlexicalized parsing. In Proceedings of ACL 41.

Lin, Chin-Yew and Franz Josef Och. 2004. Automatic evaluation of machine translation quality using longest common subsequence and skip-bigram. In In Proceedings of ACL 42.

Lu, Yumao, Fuchun Peng, Xin Li and Nawaaz Ahmed, 2006, Coupling Feature Selection and Machine Learning Methods for Navigational Query Identification, In Proceeding of CIKM.

Radlinski, F., Kurup, M. and Joachims, T. 2007. Active exploration for learning rankings from clickthrough data. In *SIGKDD*.

Salton G. and E. Voorhees. 1984. Comparison of two methods for Boolean query relevancy feedback. Information Processing & Management, 20(5).

Wilcoxon, F. 1945. Individual Comparisons by Ranking Methods. Biometrics, 1:80–83.

Xu Q. and W. Croft. 1996. Query expansion using local and global document analysis. In Proceed of the 19th annual international ACM SIGIR.

Zheng, Z., H. Zha, K. Chen, and G. Sun. 2007. A regression framework for learning ranking functions using relative relevance judgments. In Proceedings of SIGIR 30.