Learning Query Intent for Sponsored Search

Zeyu Zheng*
Peking University
No. 5, Yiheyuan Road, Beijing
P. R. China, 100080

Jun Yan
Microsoft Research Asia
Beijing
P. R. China, 100190

Chi Zhang*
Peking University
No. 5, Yiheyuan Road, Beijing
P. R. China, 100080

Dou Shen
Microsoft Corporation
One Microsoft Way
Redmond, WA, USA

Zheng Chen
Microsoft Research Asia
Haidian, Beijing
P. R. China, 100190

Ying Li
Microsoft Corporation
One Microsoft Way
Redmond, WA, USA

{v-zheng, junyan, v-chizha4, Shendou, zhengc, yingli}@microsoft.com

ABSTRACT

As sponsored search contributes the major income of many search engines, how to deliver ads effectively is studied intensively from both industrial and academia. Among various previous studies, many of them are keyword relevance based and few considered the underlying user intents behind queries for ads delivery. In this paper, we propose to classify search queries into different search intent categories such as “car buyer”, “car reviewer” and “car maintainer” about cars for ads delivery. However many classical machine learning models for query classification may fail in categorizing queries into the predefined intent categories since it is hard to collect data for training and in addition, the traditional bag of words features cannot accurately reflect user intents. In this work, we propose a random walk based solution on top of Web click-through graph, which optimize the features for measuring user intent during random walk as well. Through this way, we can automatically generate large scale training data, weight features according to query intent, and classify queries into intent categories simultaneously. The experimental results on real world user search click-through data show that our approach can improve ad click-through rate (CTR) as high as 62.5% in contrast to the relevance based ads delivery solutions without taking user intent into consideration.

Categories and Subject Descriptors
H.3.3 [Information Storage and Retrieval]: Information Search and Retrieval – search process; H.4.m [Information Systems Applications]: Miscellaneous.

General Terms
Management, Measurement, Performance, Algorithm

Keywords
User Intent, Random Walk, Sponsored Search, Classification

1. INTRODUCTION

With the rapid growth of World Wide Web, sponsored search has become one of the major advertising channels for advertisers. It represents a market of more than $11 billion [16] according to Forrester research projects. However, the classical keyword relevance based solution may fail in delivering the right ads to the end users due to the lack of consideration on the underlying query intents. For example, when two different users issue two queries “BMW X5 engine broken” and “BMW X5 engine price” to a search engine, they will see similar ads. Though both queries can well match the keyword “BMW X5”, it is obvious that the two queries have two different intents. For the user who searched query “BMW X5 engine broken”, he/she most likely wants to repair car engine, while the other user may want to buy a new one. As a result, learning to understand the user intents behind the users’ search queries is crucial for Sponsored Search.

In this paper, we propose to classify queries into different user intent categories of certain domains for improving the effectiveness of ads delivery. Through this way, the ads are displayed by not only keyword relevance, but also relevance between query intents and ads. For example, among the ads that have bid the keywords [9, 10] about cars, we can find that some of them are for car buyers, some of them are for car repairing, and some of them are for car reviewers through manual study. As a result, we may define the intent categories as car buyer, car maintainer and car reviewer in the car model domain for query classification and ads delivery.

However, even though the user intent categories could be well defined, classifying queries’ intents into these categories is much more challenging than classical query classification problems. First of all, as pointed out in [19], user intent is very hard or even impossible to be labeled by human editors. As a result, many classical supervised machine learning models, which need large scale training data, are hard to be used in this scenario. Secondly, the classical Bag of Words (BOW) [8] features for query classification may fail in reflecting the real user intent, as the feature selection and extraction algorithms for distinguishing user intents are underexplored and the effect of some good intent related features are easily dampened by large amount of general BOW features. For example, if we use the BOW features for the queries “BMW X5 engine broken” and “BMW X5 engine price”, they have three features in common while only one term is different. However, their intents are different though they are very similar in terms of term overlap in BOW model.

*This work was done when the author was visiting Microsoft Research Asia.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ADKDD ’10, July 25, 2010, Washington D.C., USA.
Copyright 2010 ACM 978-1-4503-0221-0/0...$10.00.
Motivated by those challenges, we propose a novel random walk \cite{6} based solution on the Web click-through graph to classify user queries into intents. To get the training data without the highly cost manual labeling work, we propose to manually define seed queries for each user intent category based on human common sense and then expand to more queries that have similar intents automatically. For example, the queries “buy BMW X5”, “Toyota sale” satisfy the human common sense that they want to buy cars. And we assume that “the queries with similar click behaviors may have higher probability to have the same intent.” According to this assumption, we can expand to include more queries with similar intents through random walk on the Web click-through graph starting from those manually defined queries. On the other hand, each query, which is a node in the graph, is represented as a weighted vector of terms, the weights of each term are updated automatically during the random walk and are optimized for reflecting the user intents. Through this way, the expanded queries on the Web graph are naturally classified with features weighted for reflecting query intent. As for the new queries that have not appeared in the Web graph, we can easily adopt any classical query classifier by using the expanded queries as training data and the weighted intent features for classification.

The major contributions of this paper are: (1) we propose to deliver ads according to query intent instead of keyword relevance in sponsored search; (2) we propose a novel method to classify search queries into intent categories, which breaks the bottleneck facing by many classical query classification algorithms; and (3) we conduct experiments on real user search behaviors in a commercial search engine to verify the effectiveness of our intent based sponsored search. The experiment results show that after considering query intent, the ad click through rate (CTR) can be improved as high as 62.5\% in contrast to the relevance based ads delivery solutions without taking user intent into consideration.

The rest of this paper is organized as follows. In Section 2, we introduce the previous related works. In Section 3, we formally define the user intent and mathematically formulate the problem to be solved in this paper. In Section 4, we introduce detail of the proposed user intent classification solution. In section 5, we provide the empirical study results on real user search sessions from a commonly used commercial search engine. Finally, we conclude this work in Section 6.

2. RELATED WORK

Recently, Sponsored Search (SS) is intensively studied by both industrial and academia. The history of sponsored search is well introduced in the work \cite{9}. Technically, Feng et al. \cite{10} provided a good survey on the fundamental algorithms that aim to balance the revenue of publishers and end users’ experience in sponsored search. However, as pointed out by some related studies \cite{1, 17 and 20}, the classical keyword relevance based solution cannot provide good enough results to satisfy end users and advertisers in terms of ads delivery. To improve ad click-through rate (CTR) as well as improve user satisfaction rate to ads, some recent works \cite{14, 18} proposed to leverage state-of-the-art ranking algorithms for ad ranking. In addition, many works proposed to leverage additional meta-information to help deliver more relevant ads. For instance, Broder, et al. \cite{4}, proposed a classification scheme considering four different kinds of features for ads delivery. However, few of these previous studies considered underlying intents of search queries for ads delivery, which is the major focus of this paper.

Besides the research efforts for sponsored search, Behavioral Targeting (BT) \cite{5, 21} is attracting more attention than ever before since it can analyze the user behaviors to predict what ads are more relevant to end users than others. There are many successful commercial systems using BT technologies \cite{22, 23, 24, 25 and 26}. However, the state-of-the-art BT solutions generally consider user behavioral sequence for ads delivery but in sponsored search, we interest more in what ad to display based on one single search query. This is more challenging than classical BT problem since the algorithm needs to understand the user intent only based on a short query without much context information.

Despite the related studies in the field of online advertising, a number of research works have dedicated on understanding query intents for various applications. For example, the work \cite{3} proposed to classify search queries into three intent categories, which are known as “informational”, “navigational” and “transactional”. This intent definition obviously cannot satisfy the requirements of ads delivery though it has been proved to help search engine. On the other hand, some work \cite{7} has been proposed to identify commercial intent from search queries, which can help online advertising a lot. However, even though we can identify the search queries with commercial intent, we cannot guarantee the delivered ads are relevant to the query intent.

In terms of machine learning, the query intent understanding problem is generally formulated as classification problem, and some pilot research works could be found in \cite{2, 7, 13 and 15}. However, most traditional approach relies on large amount of training data, which is hard to gain in our scenario as pointed out in \cite{19}. Moreover, the feature that can distinguish user intent is also underexplored. Differently, in this work we proposed a random walk based solution, which can generate training data, select feature and build classification model simultaneously.

3. PROBLEM FORMULATION

With the rapid growth of World Wide Web, the search engine is changing our daily lives in terms of information access. As a result, the sponsored search has become one of the major advertising channels of many advertisers. However, the state-of-the-art keyword relevance based sponsored search may fail to deliver relevant ads to users since it has little consideration to the query intent behind the keywords appear in queries. For example, if a user’s query is “BMW X5 engine price”, the intent of the user could be to check price and then to buy an engine for her car. If a user’s query is “BMW X5 engine broken”, the user most likely want to have her engine repaired. However, if we issue these two queries to a commonly used commercial search engine, we can see from Figure 1 that the ads delivered to both queries are very similar since both queries contain the same keyword “BMW X5”.

Motivated by these observations, we propose in this work to understand query intent, say, “buy” or “repair” intent in this example, to accurately deliver ads.

Since the ads are generally categorized into domains such as “ads about cars”, “ads about computers” etc. before ads delivery, in this work, we define the query intent in a domain specific manner. Mathematically, we use $D_r$ to represent an ad domain and within domain $D_r$, the query intent $I_{D_r} = \{I_{D_r}^1, I_{D_r}^2, \ldots, I_{D_r}^{l_{D_r}}\}$ is defined as a set of user tasks behind the user search queries, where $l_{D_r}$ represents the number of intent categories in domain $D_r$. Taking the car model domain as an example, through analyzing the ads in car domain of a commonly used commercial search engine, we
define that $I_{D_w} = \{\text{buy, review, maintain}\}$, which has three different query intents for ads delivery, and thus we have $I_{cap} = 3$. The ads for car sale should be delivered to the queries which have intent to buy cars; the ads for car branding should be delivered to queries either have intent to buy car or have intent to review cars; and the ads for car insurance, repair, refit should be delivered to the queries which have intent to maintain cars. Thus the ads are delivered according to the intent of the queries, but how to define the intent categories is not the focus of this paper.

In this work, our goal is to answer the question of how to classify any search query $q$ into the predefined intent categories $I_{D_w}$ under a given domain $D_w$. As introduced in previous machine learning literatures [7, 15] for query classification, the traditional classification algorithms not only depend on high quality training data but good features which can distinguish real query semantics as well. However, as pointed out in Section 2, the lack of training data and the under-exploration of the features that can distinguish query intents seriously limit the application of those traditional classification models. So, in this work, we formally define our problem as:

Given a domain $D_w$ and the predefined query intent categories $I_{D_w}$, how to train a classifier

$$C: q \rightarrow I_{D_w}$$

without large scale manually labeled training data and can generate a good set of intent based features simultaneously.

4. QUERy INTENT CLASSIFICATION

In this section, we propose a random walk based solution for query intent classification. In Section 4.1, we introduce the detail of our solution step by step. In Section 4.2, we summarize the proposed algorithm to ease the implementation of it.

4.1 The Random Walk Based Query Intent Classification

Before introducing the details of the proposed query intent classification solution, we define the terminologies to be used in the remaining part of this paper first. Suppose $G = (V, E)$ is an undirected graph learned from the click-through log of a commercial search engine, where $V$ consists all searched queries, and if two queries clicked the same page, there is an edge between the corresponding nodes, i.e.

$$E = \{(q_i, q_j) | q_i \text{ and } q_j \text{ have clicked the same page}\}$$

The weight $w_{ij}$ for edge $(q_i, q_j)$ is the number of common clicks by query $q_i$ and $q_j$. An exemplar graph is shown in Figure 2, where the phrase in pane represents an issued query and the arc represents the edge with the URL on it represents the common clicks of the two queries. On the other hand, each query $q_i$ is represented by a feature vector $f_i$, and $n$ is used to represent the number of features in the feature space. Moreover a feature weighting vector $\vec{w}^F$ is learned automatically to reweight the features for intent category $I_{D_w}^k$. When conducting the random walk process, we use $P_{\text{trans}}(i|j)$ to represent the transition probability from query $q_j$ to $q_i$, and $P_k^F(i)$ stands for the probability that query $q_i$ belongs to the intent $I_{D_w}^k$ after $t$ rounds of iteration.

![Figure 1. Similar ads are displayed for queries with different intents](image1)

(a) Query with intent to repair car engine

BMW X5 engine

Shopping Images

(b) Query with intent to buy a car engine

Figure 1. Similar ads are displayed for queries with different intents

Though it is hard or even impossible to label query intent by human editors [19], many intuitive insights, which could be formulated as a small number of search queries, can be considered as exemplar queries for each query intent, which is named as seed queries of each query intent category. For example, for the intent “buy a car”, the queries “buy BMW”, “TOYOTA on sale” etc. could all be considered as seed queries of this task since these queries satisfy the human common sense of having the intent of “buy a car”. In this paper, the seed queries we use for each intent category are selected manually, and the details could be found in Section 5. However, these manually defined queries are hard to be directly used in traditional classification models since they are seriously biased and have limited coverage.

After the seed queries are defined, we use $P_k^F(i)$ on the graph $G = (V, E)$ and initialize the feature weight vector $\vec{w}^F$, i.e.

$$P_k^F(i) = \begin{cases} 1, & q_i \text{ is a seed query} \\ 0, & \text{otherwise} \end{cases}$$

$$\vec{w}^F = \left(\frac{1}{n}, \frac{1}{n}, \ldots, \frac{1}{n}\right)$$

Using our assumption that “the queries with similar click behaviors may have higher probability to have the same intent”, we propose to update $P_k^F(i)$ iteratively according to the structure of graph $G$, which is built on the click-through information, i.e.

![Figure 2. An exemplar graph builds on click-through log](image2)
\[ p_t^k(i) = \theta_{\text{trans}} \times \sum_{q \in D_t, j \neq i} \left( \frac{p_{t-1}(j) \times p_{\text{trans}}(l|j)}{Z_j} \right) + (1 - \theta_{\text{trans}}) \times p_{t-1}^k(i) \]

where the parameter \( \theta_{\text{trans}} \) is used to balance the self-transition probability and the transition probability between queries. This formula means the probability of whether a query \( q \) belongs to intent \( I^k_{t-1} \) is determined by both the state of the corresponding query in previous iteration and the state of queries that have similar click behavior with \( q \).

When we calculate the transition probability \( P_{\text{trans}}(l|j) \), both the link information in the Web click-through graph and the content of the queries are considered in our solution. We define the transition probability from one query to the other as

\[ P_{\text{trans}}(l|j) = \frac{1}{Z_j} \left( \alpha \sum_k w_{j,k} + (1 - \alpha) \text{sim}(q_i, q_j) \right) \]

where \( \alpha \) is a parameter to balance the importance between the click-through information and the information contained in the queries. On the other hand, the \( \text{sim}(q_i, q_j) \), which is used to measure the content similarity between two queries, is defined as:

\[ \text{sim}(q_i, q_j) = \frac{\bar{g}_i \cdot \bar{g}_j}{||\bar{g}_i|| \cdot ||\bar{g}_j||} \]

Note in formula (2), \( Z_j \) is used as a normalization factor to guarantee the summation of the probabilities equals to one, i.e.

\[ \sum_{k \in V} P_{\text{trans}}(k|j) = 1, (j = 1, 2, \ldots, m) \]

Until now, we have updated \( P_t^k(i) \) iteratively utilizing the formula given above. However, as pointed out in previous sections, the direct bag of words features may fail to reflect the true query intent, and thus may mislead the calculation of \( \text{sim}(q_i, q_j) \). Consequently, we propose to optimize the feature weighting vector \( \lambda^k \) according to the result of each iteration. We define the distance between two queries in bag of words model as,

\[ \text{dis}(q_i, q_j) = ||\mathbf{f}_i - \mathbf{f}_j|| \propto (\mathbf{f}_i - \mathbf{f}_j)^T (\mathbf{f}_i - \mathbf{f}_j) \]

Suppose \( \lambda^k \) is used to reweight features to guarantee the reweighted features can reflect query distance in terms of query intent, then the query distance after feature reweighting is

\[ \text{dis}(q_i, q_j, \lambda^k) \propto (\lambda^k \cdot (\mathbf{f}_i - \mathbf{f}_j))^2 \]

Thus we aim to minimize the queries’ distances which belong to the same intent category. We summarize the optimization objective into a loss function

\[ \text{loss}(\lambda^k, Q^k) = \frac{1}{2} \sum_{i \in Q^k \setminus \lambda^0 \setminus \lambda^k} \text{dis}(q_i, q_j, \lambda^k) \]

As a summary, after initializing the probabilities in the graph \( G=(V, E) \), we can use the random walk to update \( P_t^k(i) \) through equation (1) and (2) using the default feature weights. Through this way, we can get a set of queries \( Q^k \) with intent \( I^k_{t-1} \) after each round of iteration, where \( Q^k \) is defined as

\[ Q^k = \{ q_i | P_t^k(q_i) > \theta_{\text{thres}} \} \]

The \( \theta_{\text{thres}} \) is a threshold used for query selection, so that the queries with small probability to belong to this intent will not be chosen. Utilizing \( Q^k \), we update \( \lambda^k \) by optimizing

\[ \lambda^k = \arg \min_{\lambda} \text{loss}(\lambda^k, Q^k) \]

with the constraint that

\[ \lambda^k \cdot \bar{C} = 1, \text{where } \bar{C} = (1,1,1, \ldots, 1)_n \]

The detail solution for this optimization problem is introduced in the appendix. The procedure described above is performed iteratively until it converges or reaches a predefined maximum iteration number \( T \). After the iteration stops, all queries in \( Q^k \) are naturally classified into the intent category \( I^k_{t-1} \).

Moreover, in order to classify future new queries that do not appear in the graph \( G=(V, E) \), we suppose that after the random walk process is performed for all query intent categories \( I^k_{t-1}, k = 1,2, \ldots, l_r \) in domain \( D_r \), we get \( l_r \) feature weighting vectors \( \lambda^k \) and query sets \( Q^k \). Then, for each intent category \( I^k_{t-1} \), a classical classification model such as Support Vector machine (SVM) could be trained on training data \( \cup_{k=1}^{l_r} Q^k \) and \( \lambda^k \) is used to reweight features for intent classification.

### 4.2 Algorithm Summary

**Input:** \( l_r \), sets of manually defined seed queries for all intent categories \( I^k_{t-1} \) in domain \( D_r \).

**Output:** Classified queries with selected features for classification.

**Step 1.** Build the link graph \( G=(V, E) \) based on search click-through log

\[ E = \{(q_i, q_j) | q_i \text{ and } q_j \text{ clicked the same page}\} \]

**Step 2.** Initialize the graph and feature weights:

\[ p_0^k(i) = \begin{cases} 1, & q_i \text{ is a seed query} \\ 0, & \text{otherwise} \end{cases} \]

\[ \lambda^k = \begin{cases} \frac{1}{n}, & \text{otherwise} \end{cases} \]

**Step 3.** For \( t \in [1..T] \), we calculate:

\[ p_t^k(i) = \theta_{\text{trans}} \times \sum_{j \in V, j \neq i} \left( p_{t-1}^k(j) \times P_{\text{trans}}(l|j) \right) + (1 - \theta_{\text{trans}}) \times p_{t-1}^k(i) \]

\[ P_{\text{trans}}(l|j) = \frac{1}{Z_j} \left( \sum_k w_{j,k} + (1 - \alpha) \text{sim}(q_i, q_j) \right) \]

\[ Q^k = \{ q_i | p_t^k(q_i) > \theta_{\text{thres}} \} \]

Update parameter \( \lambda^k \) according to \( Q^k \)

**Step 4.** For each \( k \in [1..l_r] \), repeat Step 2, 3, 4.
Till now, the query intent classification model is introduced in details, and we summarize it in figure 3. As we iteratively expand the initial seed queries by including more queries with the same intent, we can classify queries into predefined intent categories. If using the expanded queries as training data to classify new queries, the training dataset is automatically built and enlarged. On the other hand, through considering the click-through information, we adjust weight for each feature automatically so that the features that can reflect the real query intent will get a higher weight. In this way, we address all challenge issues introduced in previous section for classifying query intent.

5. EXPERIMENTS

In this section, we empirically show that our approach can effectively classify queries into intent categories and can improve the relevance of sponsored search. In Section 5.1 we introduce the dataset we used for experiments. And then we show the query intent classification results in Section 5.2 and finally in Section 5.3, we show how much the query intent classification can help sponsored search.

5.1 The Dataset

In the experiments, we use the click through log of a commonly used commercial search engine, which records all search queries used in it with their clicked URLs. An example of the data format could be found in Table 1, where “Query” stands for a query searched by a user, “Ads Impression” represents the ads delivered to that user and “Ads Clicked” represents the clicked ads by the user while “Page Clicked” represents the general Web pages clicked by this user using the given query.

Table 1. An exemplar click-through data sample used in this work

<table>
<thead>
<tr>
<th>Query</th>
<th>Ads Impression</th>
<th>Ads Clicked</th>
<th>Page Clicked</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://www.edmunds.com/bmw/x5/">http://www.edmunds.com/bmw/x5/</a>...</td>
<td><a href="http://www.edmunds.com/bmw/x5/">http://www.edmunds.com/bmw/x5/</a>...</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 2. The number of ads studied in this work

<table>
<thead>
<tr>
<th>Domain</th>
<th>Intent category</th>
<th>Number of Ads</th>
</tr>
</thead>
<tbody>
<tr>
<td>Car</td>
<td>Buy</td>
<td>152</td>
</tr>
<tr>
<td></td>
<td>Review</td>
<td>63</td>
</tr>
<tr>
<td></td>
<td>Maintain</td>
<td>67</td>
</tr>
<tr>
<td>Computer</td>
<td>Buy</td>
<td>144</td>
</tr>
<tr>
<td></td>
<td>Review</td>
<td>96</td>
</tr>
<tr>
<td></td>
<td>Maintain</td>
<td>41</td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td>563</td>
</tr>
</tbody>
</table>

5.2 Results for Query Intent Classification

Utilizing the dataset introduced above, we firstly empirically verify that our proposed solution can well classify queries into intents. In Section 5.2.1, we introduce the experimental configuration and then in Section 5.2.2, we introduce the experimental results. Finally we analyze the sensitivity of our proposed solution to all parameters.

5.2.1 Experimental Configuration

Baseline: to compare our proposed solution with some classical classification models for query intent classification, we consider the Support Vector Machine (SVM) [12] and Logistic Regression (LR)\(^3\) as baseline algorithms. Both baseline models are trained directly on the seed queries using two different set of features. The first one is the traditional Bag-of-Words (BOW) [8] features using the term frequency inverted document frequency (TFIDF) indexing and the other is the feature reweighted after the random walk process we proposed (RW). As a result, it yields four baseline results for comparison purpose, which are referred as “SVM+BOW”, “SVM+RW”, “LR+BOW” and “LR+RW” respectively. In addition, in order to verify the effectiveness of the feature reweighting in our proposed approach, we conduct the traditional random walk without considering the query similarity for feature reweighting, and this baseline algorithm is referred as “RW” afterwards. Moreover, our proposed approach would be referred as “QIC” (query intent classifier) in the remaining part of the experiments.

Seed queries: for each query intent, the seed queries are manually defined. However, the straightforward exemplar queries such as queries introduced in section 4 are hard to be directly used for computation, because it might be biased to some brands or entities that appear in the seed queries. For example, if we can define “buy BMW X5” has the intent to buy a car, we can see that the query “buy Toyota Camry” also has the intent to buy cars. Inspired by this observation, we first collect the product list in the studied domains and each product name is considered as an entity. Through this way, we define the seed query patterns instead of seed queries for expansion, which means we use a general word “Entity” to replace product names in queries. For example, we use “buy Entity” as seeds, where “Entity” could be any product name in our studied domain. To save place, we only give all seed patterns for category “car buyer” in Table 3 as an example, and seed patterns for other categories are defined in a similar way. Through replacing the word “Entity” by real entities such as “BMW X5”, “Toyota T-100”, etc. in seed patterns, we get the seed queries for an intent category.

\(^3\)http://www.autonlab.org/autonweb/14717.html
The precision $\theta$ used car Entity how much  Entity cheap  Entity $\textit{recall;}$ for sale  Entity, which controls the query selection threshold ents, we how much is Entity $\textit{recall}$. In this subsection, we sale Entity 0 $\times$ 0, the precision, recall and F1 is defined as $\theta$ cheapest Entity, which controls t $\theta$ discount code for  Entity 0 + = 20 $\times$ ning data and in two $\theta$ best price Entity he self buy Entity buyer performance of buy used Entity buy new Entity for sale Entity in sale how much  Entity rebate in sale best price Entity in sale how much is Entity buy used Entity used car Entity

<table>
<thead>
<tr>
<th></th>
<th>Buy</th>
<th>Review</th>
<th>Maintain</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM+BOW</td>
<td>0.71</td>
<td>0.63</td>
<td>0.78</td>
</tr>
<tr>
<td>SVM+RW</td>
<td>0.77</td>
<td>0.63</td>
<td>0.81</td>
</tr>
<tr>
<td>LR+BOW</td>
<td>0.70</td>
<td>0.66</td>
<td>0.83</td>
</tr>
<tr>
<td>LR+RW</td>
<td>0.72</td>
<td>0.68</td>
<td>0.81</td>
</tr>
<tr>
<td>RW</td>
<td>0.79</td>
<td>0.65</td>
<td>0.7</td>
</tr>
<tr>
<td>QIC</td>
<td>0.78</td>
<td>0.71</td>
<td>0.88</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Buy</th>
<th>Review</th>
<th>Maintain</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM+BOW</td>
<td>0.73</td>
<td>0.67</td>
<td>0.77</td>
</tr>
<tr>
<td>SVM+RW</td>
<td>0.66</td>
<td>0.64</td>
<td>0.65</td>
</tr>
<tr>
<td>LR+BOW</td>
<td>0.62</td>
<td>0.69</td>
<td>0.78</td>
</tr>
<tr>
<td>LR+RW</td>
<td>0.65</td>
<td>0.72</td>
<td>0.79</td>
</tr>
<tr>
<td>RW</td>
<td>0.71</td>
<td>0.71</td>
<td>0.82</td>
</tr>
<tr>
<td>QIC</td>
<td>0.77</td>
<td>0.76</td>
<td>0.83</td>
</tr>
</tbody>
</table>

| Entity buy | buy Entity |
| Entity cheap | cheap Entity |
| Entity sale | sale Entity |
| Entity for sale | for sale Entity |
| Entity lowest price | cheapest Entity |
| Entity price | cheapest prices Entity |
| Entity rebate | discount code for Entity |
| Entity dealer | best price Entity |
| Entity in sale | how much Entity |
| buy new Entity | how much is Entity |
| buy used Entity | used car Entity |

Evaluation Metric: in this work, the precision, recall and F1, which are three of the most commonly used evaluation metrics for classification problems [11] are used for evaluation. When predicting intent $I_{D_s}^k$, the precision, recall and F1 is defined as

Precision $= \frac{\text{number of queries correctly classified as } I_{D_s}^k}{\text{number of queries classified as } I_{D_s}^k}$

Recall $= \frac{\text{number of queries correctly classified as } I_{D_s}^k}{\text{number of queries having intent } I_{D_s}^k}$

$F1 = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$

Parameter settings: In our experiments, we manually set the parameters as $\theta_{\text{trans}} = 0.2$, $\theta_{\text{thres}} = 0.4$, $\alpha = 0.5$ and $T = 20$.

5.2.2 Results
To show the effectiveness of our proposed algorithm in contrast to the baseline algorithms, we conducted experiments in two different domains, i.e. the car model domain and the computer product domain and the results are shown in Table 4 and Table 5 respectively. From those tables, we can see that our approach can consistently give best performance in terms of F1. In the car model domain, the improvement could be as high as 7% while we can improve F1 as high as 5% in the computer domain.

From the baseline “SVM+BOW” and “LR+BOW”, we can see that no matter which baseline classification model is used, due to the lack of training data and the unrepresentative features, classical models may fail to classify query intent effectively. However, on one hand, for baseline “SVM+RW” and “LR+RW”, which use the features reweighted by our algorithm, outperform the basic ones, i.e. “SVM+BOW” and “LR+BOW”. On the other hand, the traditional random walk algorithm, which only concerns the click-through information and does not rely on large scale training data, also performs better than other baselines. As a result, we can conclude that our approach can effectively improve the quality of training data, which is crucial for classification models.

5.2.3 The Parameter Analysis
In section 5.2.2, the effectiveness of our approach is empirically studied. However, the sensitivity of the algorithm to the parameters $\theta_{\text{trans}}$, which controls the self-transition probability; $\theta_{\text{thres}}$, which controls the query selection threshold; $\alpha$, which balance the importance of the click-through information and query similarity; and $T$, which indicates the number of iterations, have not been discussed. In this subsection, we conduct the sensitivity analysis to those parameters in our approach empirically. To save space, all experiments in this subsection are conducted in intent category “car buyer” under the car model domain.

We first fix other parameters to test the performance of the proposed QIC against the number of iterations $T$. From Table 6, we can see that as iteration continues, the recall increases while the precision decreases. This is because in each round of iteration,
we include more queries, which have high possibility to have the intent to buy. However, there is still a chance that some of those queries do not have this intent. As a result, the precision decreases. But anyhow, including new queries can more or less enlarge the coverage, thus lead to the increase of recall and the performance as well in terms of F1.

Table 6. Performance of QIC against the number of iterations

<table>
<thead>
<tr>
<th>$T$</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.95</td>
<td>0.12</td>
<td>0.21</td>
</tr>
<tr>
<td>5</td>
<td>0.90</td>
<td>0.28</td>
<td>0.43</td>
</tr>
<tr>
<td>10</td>
<td>0.87</td>
<td>0.31</td>
<td>0.46</td>
</tr>
<tr>
<td>15</td>
<td>0.81</td>
<td>0.46</td>
<td>0.59</td>
</tr>
<tr>
<td>20</td>
<td>0.78</td>
<td>0.51</td>
<td>0.62</td>
</tr>
<tr>
<td>25</td>
<td>0.78</td>
<td>0.51</td>
<td>0.62</td>
</tr>
</tbody>
</table>

Then in Figure 4, we study the influence of parameters $\theta_{\text{trans}}$, $\theta_{\text{thres}}$ and $\alpha$ in terms of F1. As all those parameters may change from 0 to 1, we plot all the 3 lines in one figure. From that figure we can see that with the change of any parameters, our algorithm can consistently give good performance especially when they fall in the space [0.2, 0.5]. Motivated by this observation, we set $\theta_{\text{trans}} = 0.2, \theta_{\text{thres}} = 0.4$ and $\alpha = 0.5$ in our paper.

Figure 4. The performance of QIC against parameter $\theta$

5.3 Results for Query Intent Based Ads Delivery

In this subsection, we empirically verify that through considering the query intent, we can significantly improve the effectiveness of ads delivery. In Section 5.3.1, we first give our evaluation metrics. Then in Section 5.3.2, we show and analyze the results. Finally in section 5.3.3, we give a case study of how we deliver ads according to our approach.

5.3.1 Evaluation Metrics

The performance of Sponsored Search is generally evaluated by two metrics, which are ad click-through rate (CTR) and conversion rate. Though the latter is important for advertisers, it is hard for us to track it from the log due to privacy issues. As a result, CTR is chosen as one of the evaluation metrics. However, a higher CTR along is not sufficient to conclude that a algorithm is better than other solutions, because it only guarantees the precision of the delivered ads and has little consideration on the coverage of the ads delivery. Motivated by this, we propose to adopt the classical evaluation metrics F1 [11] to evaluate the effectiveness of the Sponsored Search. For an ad $a$ and a set of queries $Q^*$, we define “precision”, “recall” and “F1” as

$$\text{precision}(a, Q^*) = \frac{\#\text{number of queries in } Q^* \text{ clicked } a}{\#\text{number of queries in } Q^*} = \text{CTR}$$

$$\text{recall}(a, Q^*) = \frac{\#\text{number of queries in } Q^* \text{ clicked } a}{\#\text{queries clicked } a}$$

$$F1(a, Q^*) = \frac{2 \times \text{precision}(a, Q^*) \times \text{recall}(a, Q^*)}{\text{precision}(a, Q^*) + \text{recall}(a, Q^*)}$$

From the definition, it is easy to conclude that the larger the precision is, the more accurate ads are delivered; the larger the recall is, the better coverage it has; and the larger the F-measure is, the better balanced performance we can achieve.

In the following experiments, the algorithm that only considers the keyword relevance score is performed and referred as “Baseline”. On the other hand, we use “Manual” to represent the ads delivery strategy based on manually labeled query intents. To save place, we do not show other baseline algorithms introduced in section 5.2.
5.3.2 The Result
In this subsection, using the evaluation metrics introduced above, we show the effectiveness of delivering ads according to the query intent through experimental results in car model domain and computer domain. The results are shown in Table 7 and Table 8 respectively. From those two tables, we can see that our approach can always outperform the baseline algorithm, this is because our approach not only considered the keyword relevance, but the underlying query intent as well, which can help search engine understanding what information an end user really wants. Especially when delivering ads which want to sell cars (intent “buy” in car model domain), we can improve the precision (CTR) as high as 62.5%, while only lose a little recall. Moreover, we can see from those tables that using the manually classified query intents, we can achieve relatively high performance, which indicate that it is reasonable to deliver ads according to the query intent.

5.3.3 A Case Study
As given in previous sections, when two different users issue two different queries “BMW X5 engine broken” and “BMW X5 engine price”, the ads delivered by a commonly used commercial is shown in Figure 1, which would be regarded as having intent of buy. However, the query “BMW X5 engine broken” would be classified as having the intent of maintain in our approach. As a result, instead of delivering ads that have intent of buy, we would deliver ads that focus on car maintaining, which we believe is more relevant to the issued query considering the intent. More sample cases are shown in Table 9, and from that table, we can confidently conclude that the intent based ads delivering strategy can really help search engines to deliver ads.

<table>
<thead>
<tr>
<th>query</th>
<th>Intent</th>
<th>Method</th>
<th>Ads Delivered (title)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Good sports car picture</td>
<td>Review</td>
<td>Relevance based</td>
<td>Audi Sports Cars</td>
</tr>
<tr>
<td>Repair store Toyota T-100</td>
<td>Maintain</td>
<td>Relevance based</td>
<td>Toyota T100 car Store</td>
</tr>
<tr>
<td></td>
<td>Intent based</td>
<td></td>
<td>Toyota T100 Repair Manual</td>
</tr>
</tbody>
</table>

In the future, we plan to apply our approach to more domains and test it in an online environment to check whether this strategy can help enhance the revenue. On the other hand, the convergence of the proposed algorithms is not analyzed theoretically in this work. As a result, we propose to further analyze the algorithm in the future, and theoretically prove that the algorithm will converge after a certain rounds of iteration are performed.

7. REFERENCES
8. Appendix: $\lambda$ calculation:

Given $\lambda \cdot \hat{C} = 1$, where $\hat{C} = (1,1,\ldots,1)_n$, we optimization $\lambda$ to minimize the loss function.

Firstly, we define

$$h(\lambda, Q) = \text{loss}(\lambda, Q) - k(\lambda \cdot \hat{C} - 1)$$

Because $\lambda \cdot \hat{C} = 1$, we have $h(\lambda, Q) \equiv \text{loss}(\lambda, Q)$. So we only have to find the $\lambda$ that can minimize $h(\lambda)$.

Using lagrangian, let $\frac{\partial h(\lambda, Q)}{\partial \lambda} = 0$ and $\frac{\partial h(\lambda, Q)}{\partial k} = 0$, we have:

$$\frac{\partial h(\lambda, Q)}{\partial \lambda} = \sum_{i \in Q, j \notin Q, ij} (\lambda \cdot (f_i - f_j))(f_i - f_j) - k\hat{C} = 0,$$

$$\frac{\partial h(\lambda, Q)}{\partial k} = 1 - \lambda \cdot \hat{C} = 0$$

As a result, for $x \in [1..n]$, we have:

$$\sum_{i \in Q, j \notin Q, ij} (\lambda \cdot (f_i - f_j))(f_i - f_j) - k = 0$$

$$\sum_{i \in Q, j \notin Q, ij} \lambda_i (f_i - f_j)) = 0$$

$$\lambda_i = 1$$

From (2) and (3), we can get $n+1$ equation with $n+1$ variables, where $n$ is the number of features and it would not be very large after traditional feature selection algorithm is performed. However, as these equations may not have an analytical solution, we leverage the power of least squares estimator (LSE) to estimate the parameters $\lambda$ and $k$. Let

$$\beta = (\lambda_1, \lambda_2, \ldots, \lambda_n, k)^T$$

$$A_{ij} = f(x) = \begin{cases} 
\sum_{a,b \neq a} 2(f_{ai} - f_{bi})(f_{aj} - f_{bj}) & i,j \in [1..n] \\
1, & i = n + 1, j \in [1..n] \\
-1, & i \in [1..n], j = n + 1 \\
0, & i = n + 1, j = n + 1 \end{cases}$$

$$X = (0,0,\ldots,0,1)^T_{n+1}$$

So we only need to solve:

$$\beta = \arg \min_{\beta} \| A\beta - X \|^2$$

Since LSE solution always exists, we can always calculate $\hat{\lambda}$ from $\beta$. Thus the $\hat{\lambda}$ could be optimized through this LSE computation.