Matching Millions of Queries with Billions of Bid Keywords

ABSTRACT

Sponsored search is the main source of revenue of a search engine. Advertisers bid on phrases for their advertisements, and search engines match user queries to relevant ads based on the bid phrases. Since both queries and bid phrases are short texts and cannot be modeled by the standard bag-of-words approach, most existing approaches exploit user behavior data (e.g., click-through data, session data, etc.) to fill the semantic gap in matching bid phrases and user queries. This approach however cannot handle tail queries which do not have much user behavior data. Although individually rare, tail queries as a whole account for a considerable portion of the query volume and are a significant source of search engine's revenue. In this paper, we propose a novel approach for matching queries and bid phrases. Leveraging a probabilistic taxonomy and a large scale term co-occurrence network, we conceptualize a short text to a set of relevant concepts. To handle millions of queries and billions of bid phrases, we create a semantic index of concepts: relevant bid phrases are selected for a given query by measuring their similarity in the concept space. We conduct a series of experiments including expanding 30 million queries with 0.7 billion phrases. Experiment results show the effectiveness of our approach.

Categories and Subject Descriptors

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

General Terms

Algorithm, Experimentation

Keywords

Sponsored advertising, query expansion, keyword suggestion, text modeling

1. INTRODUCTION

CIKM '13 San Francisco, CA USA

Sponsored search aims at matching search queries to relevant advertisements. Each ad is characterized by a list of bid keywords which are representative for it. If the matching option is *exact match*, an ad has opportunity to be triggered if the given query is identical to one of its bid keyword. Another option is *smart match* which is the default option provided by mainstream search engines. In this scenario, search engines display ads whose bid keywords are semantically relevant to the given query. Smart match can potentially channel a larger traffic volume to advertisers [23]. However, in most cases, this traffic is less targeted compared with exact match. Our paper deals with finding matching bid keywords for a qeury and focus on the semantic relevance between sthem.

Both queries and bid keywords are very short (the average length is between 2.4 and 2.7 words [10]). Thus matching based on syntactic similarity suffers from low recall [23]. As a result, only 30%-40% of search queries are covered by ad results [9]. To address such serious word mismatching problem, matching must be semantic. Topic models [7, 13] express semantics as distribution over latent topics. They are useful for corpus of normal documents, but queries as well as bid keywords are too short to provide statistically meaningful signals. A more promising approach is to augment queries with additional external knowledge such as user behavior data [12, 10, 14]. However, they are only effective for popular queries, but not for *tail queries* that have little or no historical user behavior data. Unfortunately, tail queries, though individually rare, make up a significant portion of the query volume [9].

No matter a query is popular or rare, human beings can understand it easily. This is because knowledge in a human mind makes up for the sparsity of the input. If we can provide such knowledge to machines then they can better understand short texts. Gabrilovich et al [15] proposed a novel approach known as *explicit semantic analysis (ESA)*, which models a short text by a set of Wikipedia [2] concepts (it considers the title of any Wikipedia article as a Wikipedia concept). However, such taxonomy-based approach relies on the quality and coverage of the taxonomy [11] and the concept space of Wikipedia is limited (about 111,654 concepts [24]). Besides, the transformation from bag-of-words to bag-of-concepts in ESA is based on co-occurrence between terms of a short text and terms in Wikipedia articles. However, most terms in an article do not have *isA* relationships with the corresponding Wikipedia concept. Therefore, associating short texts with their relevant concepts via this

co-occurrence information is not precise. We regard the co-occurrence information used for *conceptualization* in their approach as *loose isA* relationship. For a short text, its limited number of terms makes the loose isA relationship more vulnerable.

In our approach, we expand a sparse, noisy and ambiguous input to a rich, explicit, and accurate representation via *strict isA* relationships provided by probabilistic taxonomies such as Probase [24] and Yago [22]. Then we perform similarity calculation on that representation. Our conceptualization algorithm associates each short text with its relevant concepts by inferencing technique based on probabilities provided by adopted taxonomy. We also mine frequent sense-patterns from the adopted co-occurrence network and filter inappropriate concepts assigned to short texts to avoid additional noise.

Besides of semantic problem, scalability is critical for any sponsored search approaches. Moreover, in our approach, each short text is mapped into concept space and has a distributed representation. Thus matching between short texts over our enriched representation is more complicated compared with the matching between bags-of-words (around 2 words). Comparing millions of queries with billions of keywords by a naive algorithm is intractable even in our map-reduce cluster. Thus, instead of assigning a score to each bid keyword and ranking the entire corpus, we leverage locality-sensitive hashing (LSH) to efficiently select a small set of relevant bid keywords before ranking.

In summary, we make the following major contributions:

- We propose a new approach for query expansion. Our approach is much more *robust* because: (1) We leverage web-scale, data-driven knowledgebases, and (2) we conceptualize short texts through strict isA relationships which are more reliable than through loose isA relationships.
- We propose a sophisticated algorithm to conceptualize short texts. Our algorithm resolves word mismatching problem by "lifting" the representation of short text to the concept level. Besides, it mines frequent sense-patterns to eliminate ambiguity of short text.
- Our approach can match both head and tail queries to relevant bid keywords. This is very critical for current sponsored search systems since selecting relevant ads for tail queries is the major bottleneck.
- Our approach can scale to massive datasets. We apply our approach to expand 30 million queries with 0.7 billion bid keywords for a commercial search engine.

The rest of this paper is organized as follows. In the next section, we summarize related work. We introduce the external resources (knowledgebases) used in our approach in Section 3. In Section 4, we give a brief introduction to our system architecture. We present our conceptualization algorithm in Section 5. In Section 6, we describe the process of selecting and ranking candidates to derive final results. Experiments are discussed in Section 7. We conclude our paper in Section 8.

2. RELATED WORK

Techniques proposed in this paper are related to keyword suggestion and query expansion.

Because the average length of queries are around 2 [6], query expansion has been thought as an effective way to resolve word mismatching problems [12]. Existing approaches mainly focus on expanding queries with various external resources including organic search results [8, 19], user behavior data [12, 10, 14] and bidding relationships between phrases and ads [23]. Ricardo et al [6] propose to represent queries by aggregation of terms-weight vectors of their clicked URLs, which leverages both organic search results and user behavior data. However, in most cases, there is no adequate user behavior data for tail queries that are individually rare but make up a significant portion of the query volume [9].

More fundamentally, the key problem of many queries processing tasks and keyword research is measuring similarity between short texts. Previous work on measuring text semantic similarity has focused mainly on either large documents or individual words [20]. Since both queries and bid phrases are too short to draw reliable statistical conclusion, LSA approaches [7, 13, 17] are not an effective way to index short text snippets. Besides corpus-based approaches, existing approaches for measuring short text semantic similarity are largely knowledge-based [15, 18, 11, 21]. Knowledgebase including WordNet [3], open directory project(ODP) [1] and wikipedia[2] are exploited to enrich short text snippets at semantic level. As Chen et al [11] pointed, the performance of knowledge-based approaches are strikingly affected by the accuracy and the coverage of used concept hierarchy. Our approach model and rank short text with the help of a probabilistic taxonomy named Probase [24] which, to the best of our knowledge, contains the largest concept space compared with those knowledgebases adopted by existing approaches. Besides, most of these knowledge-based approaches map short text snippets into their concept space based on co-occurrence of terms within the content of original text and terms appearing in corresponding articles or web pages of concepts. Considering that queries and bid phrases are very short, such strategy is not reasonable.

3. THE KNOWLEDGEBASE

The emergence of large scale, probabilistic knowledgebases, such as Probase [24] and Yago [22], brings new opportunities to text understanding. We adopt Probase, a probabilistic taxonomy for conceptualization in our approach. The reason we prefer Probase is that we consider a huge concept space indispensable to understand open-domain bid phrases. Probase is rich enough to cover a large proportion of concepts about worldly facts. The version of Probase we use contains about 8.26 million instances (e.g., "windows phone 7", "ipad", "kindle", etc.) and about 2.7 million concepts (e.g., "platform", "device", "ebook reader", etc.). In the rest of this paper, we use e to denote an instance, c as a concept.

All instances and concepts are organized hierarchically by the *isA* relationships. For example, "kindle" is an instance of concept "ebook reader". Note that *isA* relationships also exist between sub-concepts and concepts (e.g., sub-concept "ebook reader" is an instance of concept "device"). In contrast to traditional taxonomies that regard knowledge as black and white, Probase maintains probabilistic *isA* relationships:

$$\Pr(e|c) = \frac{n(e,c)}{n(c)} \qquad \Pr(c|e) = \frac{n(e,c)}{n(e)} \tag{1}$$

where $n(c)$, $n(e)$, $n(e, c)$ denote the frequency of c, the frequency of e, the frequency of e and c occuring together in all Hearst patterns [16] extracted from billions of documents and web pages. These probabilities have some natural properties: $P(e|c)$ reflects the typicality of e given c, and $P(c|e)$ reflects the typicality of c given e. We use these probabilities as fundamental building blocks for probabilistic models to conceptualize short texts.

We also use Probase's instance-instance co-occurrence network wherein each node is a Probase instance and each edge between two nodes is the co-occurrence relationship between instances. For 8.26 million instances, we extract their co-occurrence relationships from 1.68 billions of web pages. Each time two instances co-occur within one sentence of those web pages, we will increase their co-occurrence frequency by 1. Finally, we got a weighted network with 28.6 billion edges. After some filtering (such as removing edges with frequency ≤ 5), the ultimate network contains about 3.73 billions of edges. We will use this co-occurrence information to disambiguate multiple concepts of an instance to avoid enriching instances with noisy concepts.

4. SYSTEM FRAMEWORK

In this section, we present the system diagram (Figure 1), which shows how data flows through our system end-to-end. Just like classical IR systems, it can be divided into online and offline components.

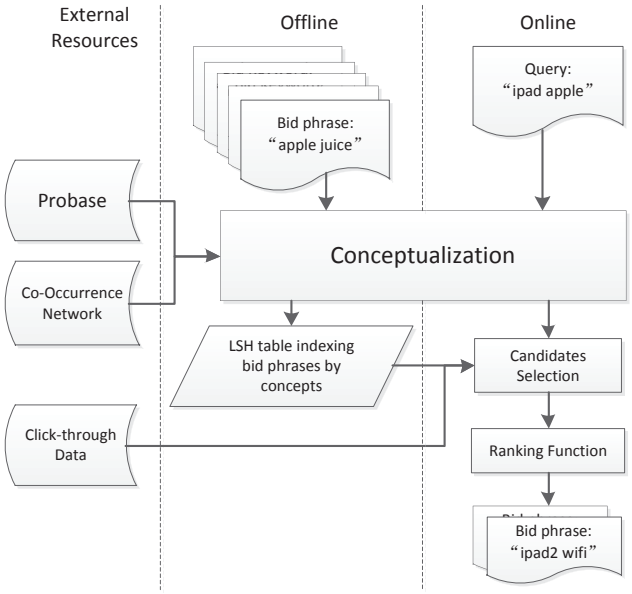


Figure 1: Overall System Architecture

In offline processing, we conceptualize each bid phrase, that is, we map each bid phrase to a set of representative concepts. This enables us to estimate the similarity between two short texts by measuring the similarity between their corresponding sets of concepts. For the purpose of large scale similarity computation, we exploit the locality-sensitive hash (LSH) scheme to index bid phrases. LSH enables us to efficiently find bid phrases whose corresponding concept sets are likely to be similar to that of the query. Besides, we also mine click-through data to extract semantic associations (i.e., co-click relationships) between queries and index them for efficient real-time looking up. At runtime, when a query is posed, we conceptualize it in the same way as we did for bid phrases. Then, from the massive bid phrase dataset, we select a small set of bid phrases that are relevant to the given query using user behavior data (for head queries) or LSH (for tail queries). Finally, we rank these candidates and use the top-ranking ones as query suggestions (expansions).

5. CONCEPTUALIZATION

Conceptualization generalizes input to concepts. It is one of the most basic cognitive process for human beings. For instance, we generalize "hot dog" to the concept of "food", and "dog" to the concept of "animal". To allow machines to handle such text understanding tasks, we exploit rich conceptual knowledge provided by Probase that covers most worldly facts.

5.1 Instances Detection

Given a short text such as "windows phone app", We firstly identify all Probase instances that appear in the short text—"windows", "windows phone", "phone app" and "app". Next, we remove an instance if it is entirely covered by another one (that is, it is a substring of another instance). Thus after this step, our example short text is transformed into a set of Probase instances {"windows phone", "phone app"}. In fact, an instance entirely covered by another instance is very likely to be either its modifier or its head that has a more general meaning. Because conceptualization "lifts" the representation of short text from term-level to concept-level, more specific instances are better for avoiding "over-abstracting".

5.2 Senses Derivation

Sense of a word or phrase is its meaning. To represent one sense of an instance, we use a set of related concepts. See Table 1 for example, the set of all "ipad"'s concepts $\{c|\ \Pr(c|\text{"ipad"}) > 0\}$ represents its only sense. However, ambiguous instances such as "apple" has more than one sense and associates totally unrelated concepts (e.g., "fruit" and "company") in Probase. We use an offline process to partition each instance's concepts into one or more clusters where each cluster consists of related concepts and thus represents one sense of the instance.

For each instance e, we cluster its associated concepts by exploiting *isA* relationships and typicality. Given an instance e, we first rank its concepts by typicality (i.e., $\Pr(c|e)$) and select the top-K most typical concepts for e. The value of K is usually in the order of tens (in our implementation $K = 15$). Then, we cluster the K concepts agglomeratively. At the beginning, each of the K concepts is regarded as an independent cluster. During the procedure, if there exists isA relationship between any two concepts, their clusters are merged.

Indeed, mainstream taxonomies (either automatically generated or manually annotated) are rather clean but not complete enough. For example, there is no isA relationship between the concept "conventional input device" and any other concepts in the first sense (cluster) of "mouse" (see Table 2). However, we can eventually merge them correctly by a simple rule: if a certain concept is the suffix of another concept, merge their clusters together. Actually, "device" is the head

of not only "conventional input device" but also most concepts in the first sense of "mouse".

This procedure results in several initial clusters. Table 2 shows initial clusters of some ambiguous instances. Let the instances of a cluster be the union of its concepts' instances, we can assign other concepts associated with e to its closest cluster where "closeness" is measured by the proportion of overlapping instances of them.

5.3 Senses Disambiguation

Suppose we detected instances {"ipad", "apple"} from short text "ipad apple" and the clustering results of these two instances are as Table 1 shows. Our goal is to identify the "company" sense as "apple" refers to in this short text. Formally, we define the task of senses disambiguation as follows:

Definition 1. Given a set of instances $E = \{e_1, \ldots, e_D\}$ where each instance is associated with one or more senses, e.g., $e \in E$ has l senses $s_1, \ldots, s_l$. We define a sense-pattern as a vector $\mathbf{p} = (p_1, \ldots, p_D)^{\mathrm{T}}$ where p_i indicates that e_i refers to its p_i-th sense in this context. Senses disambiguation is to find the correct sense-pattern.

Given an ambiguous instance such as "apple", without any context, it is impossible to judge which sense of it is expressed. However, if the instances detected from short text are "apple" and "microsoft", we know that it refers to "company" instead of "fruit". For each pair of instances e and e', suppose e has senses $s_1, \ldots, s_m$ and e' has senses $s'_1, \ldots, s'_n$, we compare each pair of their senses. We measure the similarity between s_i and s'_j by Jaccard Similarity: $\mathrm{JS}(s_i, s'_j) = \frac{|s_i \cap s'_j|}{|s_i \cup s'_j|}$. Suppose the pair (s_i, s'_j) achieves the maximum Jaccard similarity among all pairs of senses and the similarity exceeds our pre-defined threshold, we reserve s_i, s'_j for e, e' respectively and eliminate all other senses of the two instances.

However, two instances that share similar senses may not appear within one short text at the same time. For instance, "ipad apple" has three possible sense-patterns, "device-tree", "device-fruit" and "device-company". Intuitively, "company" and "device" have stronger *contextual-continuity* than the other sense-patterns. However, "company" and "device" are two totally different senses, and the Jaccard similarity of them is certainly below our pre-defined threshold.

From our dataset, we observed that there are many pairs that fit the sense-pattern "company-device", and the instances in the pairs are not ambiguous, for example, "amazon kindle", "microsoft surface", "google nexus", etc. Besides, "amazon", "microsoft" and "google" are all representative instances of the sense "company" and "kindle", "surface" as well as "nexus" are all representative instances of the sense "device". Based on the above observation, one immediate intuition is that two senses have strong contextual-continuity their representative instances have high co-occurrence. Thus, we measure contextual-continuity between two senses using our instance-instance co-occurrence network mentioned in Section 3. Let us denote the network as a weighted directed graph G where each node denotes an instance and the weight $w_{(u,v)}$ of edge (u, v) denotes the co-occurrence frequency of instances u and v. We also denote the occurrence frequency of instance u by $w(u) = \sum_{v \in G} w(u, v)$ and the total number of counted occurrences by $W = \sum_{u \in G} w(u)$. First, suppose an instance e has senses $s_1, \ldots, s_m$. For each sense $s_i, i \in \{1, \ldots, m\}$, we associate it with a set of instances (denoted by U_i). Elements of U_i are top-K nodes $u \in G$ when ranked by $\Pr(u|e, s_i)$. We compute the probability with the following independence assumptions for simplification:

$$\Pr(u|e, s_i) \propto \Pr(u) \Pr(e, s_i|u) = \Pr(u) \Pr(e|u) \prod_{c \in s_i} \Pr(c|u) \tag{2}$$

where $\Pr(u)$, $\Pr(c|u)$ are directly given by popularity and typicality and $\Pr(e|u)$ is estimated by $\frac{w(u,e)}{w(u)}$. By normalization, we have $\Pr(u|e, s_i)$ for each element $u \in U_i$. The last column of Table 1 shows the top-ranking nodes of the different senses of "apple". As you can see, they are representative instances of corresponding sense. Thus, given two instances e and e', having senses $s_1, \ldots, s_m$ and $s'_1, \ldots, s'_n$ respectively, we can measure the *contextual-continuity (CC)* between one pair of their senses s_i and s'_j by measuring the connectivity between U_i and U'_j. First, we weight the connectivity between one pair of nodes (u, v) by

Definition 2.

$$\mathrm{C}(u, v) = w(u, v) \log(\frac{W}{w(u)}) \log(\frac{W}{w(v)}) \tag{3}$$

Our weighting technique is like the famouse TF-IDF where the co-occurrence frequency $w(u, v)$ is like the term frequency (TF) and the other terms function as inverse document frequency (IDF). Then we compute the connectivity between U_i and U'_j based on the connectivities between their nodes:

Definition 3.

$$CC(s_i, s'_j) = \sum_{(u,v), u \in U_i, v \in U'_j} \Pr(u|e, s_i) \Pr(v|e', s'_j) \mathrm{C(u,v)} \tag{4}$$

We select the correct sense-pattern according to Eq.(5)

$$\mathbf{p}^* = \arg\max_{\mathbf{p}} \sum_{i=1}^{D} \max_{j \neq i} CC(s_{p_i}, s_{p_j}) \tag{5}$$

We disambiguate many senses of detected instances according to $\mathbf{p}^*$ and denote the chosen senses as $(s_1, \ldots, s_D)$. Then we construct the given short text's corresponding concept set as $\cup_{i=1}^{D} s_i\}$. By conceptualization, we eventually "lift" short text from bag-of-words to bag-of-concepts. Although enriching a short text into a set of many concepts is a big step, it hasn't made full use of the probabilistic taxonomy (i.e., Probase's typicalities). Thus, we also assign one concept vector $\mathbf{c}_i$ to sense s_i where each component of $\mathbf{c}_i$ corresponds to one concept in Probase and suppose the row r corresponds to concept c, then the value in the row r of $\mathbf{c}_i$ is $\Pr(c|e_i)$ if $c \in s_i$. Otherwise, it is zero. Besides of the concept set, we also assign the short text a concept vector: $\sum_{i=1}^{D} \mathbf{c}_i$.

6. RETRIEVAL

Besides of a collection of 0.7 billion bid phrases $P = \{p\}$ provided by a commercial search engine, we extract a collection of 30 million queries $Q = \{q\}$ from search logs. We offline expand Q using P (i.e., for each $q \in Q$, select several most related bid phrases from P). Then we build index mapping each q to its resulting expansions. When a query is

Instance	Sense	Representative instances
e_1 =ipad	s_1 = {tablet device, iso device, apple device, platform, device, mobile device, portable device, technology, tablet device, tablet, gadget, ...}	U_1 = {iphone, mobile phone, ipod, laptop, ipod touch, pdas, smartphones, apple tv, apple's ipad, phone, notebooks, kindle, ...}
e_2 =apple	s_1 = {fruit tree, crop, tree, ...}	U_1 = {peach, mango, pear, cherry, banana, carrot, potato, pecan, sweet potato, ...}
	s_2 = {fruit, food, fresh fruit, flavor, juice, snack, healthy snack, ...}	U_2{grape, banana, orange, strawberry, pear, peach, mango, cheese, cherry, chocolate, ...}
	s_3 = {company, brand, firm, corporation, ...}	U_3 = {microsoft, ibm, sony, dell, motorola, google, intel, hp, nokia, cisco, ...}

Table 1: Conceptualization of short text "ipad apple"

apple	blackberry	palm	bass	mouse
fruit	fruit	plant	warm water fish species	cursor control
food	berry	tree	fish	device
fresh fruit	small fruit	plant	predator	input device
flavor	food	*****	fish species	peripheral usb device
juice	dark berry	oil	*****	peripheral device
snack	fresh fruit	vegetable oil	wood	conventional input device
healthy snack	plant	tropical oil	*****	external device
*****	dark fruit	brand	instrument	computer peripheral
company	*****	edible oil	musical instrument	computer input device
brand	device	plant oil	stringed instrument	*****
firm	mobile device	*****	acoustic instrument	mammal
corporation	smartphones	device	sound	animal
*****	smart phone	personal digital assistant	traditional instrument	small animal
fruit tree	platform	pdas		mammalian cell
crop	phone	handheld computer	*****	model organism
tree	handheld device	platform	beer	rodent

Table 2: Initial Clusters of Ambiguous Instances

issued by user, search engine look up the index, if the given query is in that index, its expansions (related bid phrases) are used to trigger ads for this query. We also build up an online query expansion system, so when the submitted query is unseen (not in Q), we retrieve related bid phrases for it through our online system.

No matter a query is processed offline or online, we divide the retrieval procedure into two phases. First, we find a small set of candidate bid phrases (somehow relevant) using some semantic aware index. Second, we rank candidates by measuring their similarity with the given query and return the top-K bid phrases. On the one hand, we want the set of candidates to be small enough so that computational cost for ranking is minimized. On the other hand, the candidates should cover adequate relevant bid phrases.

6.1 Click Data based Retrieval

We use click through data to help find relevant queries for a given head query. User clicks define strong associations between queries and URLs [14]. Candidate selection using click data (CSCD) is motivated by the assumption that queries leading to clicks on the same URLs are semantically relevant. We obtain co-click data $\{(q, q', u, f)\}$ where f is the number of times both q and q' lead users to click u. We then build inverted index for the co-click data. Given a query at runtime, we look it up in the index. Assume the query exists in the index. For each of its co-clicked queries, we check if it is a bid phrase using *exact match*. If the query does not exist in the index (it is a tail query), we turn to our second approach for *smart match*.

6.2 Concept Set based Retrieval

By conceptualization, we transform each short text (either a query or bid phrase) into a set of concepts. Thus, we measure semantic similarity between two short texts by measuring similarity between their corresponding concept sets and do not distinguish a short text from its corresponding concept set from now on.

We adopt Jaccard similarity ($\text{JS}(p, q) = \frac{|p \cap q|}{|p \cup q|}$) and set a threshold $t = 0.8$ so that a bid phrases whose Jaccard similarity with respect to the given query is no less than t qualifies for being the candidates of that query. Thus, we can formalize our task as:

Definition 4. Given a collection of bid phrases P, a collection of queries Q, a similarity function ($\text{JS}(\cdot, \cdot)$), a threshold $t = 0.8$, to find all pairs of short texts $\{(p, q)|p \in P, q \in Q, \text{JS}(p, q) \geq t\}$

A naive algorithm needs $O(|Q| \cdot |P|)$ comparisons (exactly computing Jaccard similarity) where $|P|$ is of billion order in our setting. A popular solution is to convert Jaccard similarity constraint ($\text{JS}(p, q) \geq t$) into an equivalent overlap

constraint ($O(p,q) \geq \frac{t}{1+t}(|p|+|q|) = \alpha$) and build inverted index mapping each concept c to a list of bid phrases that contain c. We scan each query $q \in Q$, probing the index using every concept of q, and obtain a set of bid phrases; merging these bid phrases together gives us their actual overlap with q, final results can be extracted by removing bid phrases whose overlap with q is less than t. However, some concepts contain extremely long inverted lists which incur significant overhead for building and accessing them. Moreover, according to our experiment, these popular concepts can't be treated as "stop words" and thus have to be reserved. For large scale efficient similarity join, we adopt PPJoin [25] which proposed several filtering rules allowing us to scan only part of a concept's inverted list, to prune a considerable number of bid phrases in the inverted lists before actually computing Jaccard similarity. Thus, the computaional effort for exactly comparison is significantly reduced.

Constructing online query expansion system means for a query, we must select its most related bid phrases from P within tens of millisecond. Besides of the scalability of P, the not small number of elements of a concept set is also a challenge. For a given query q, to efficiently select candidates $\{p|p \in P, \mathrm{JS}(p,q) \geq t\}$, we give up those exact algorithms that have to merge hundreds of posting lists of inverted index and adopt locality-sensitive hash (LSH) which is *approximate* algorithm for finding similar items.

We represent a concept set by a 0-1 vector where each component corresponds to one concept of Probase. There is a 1 in row r when the corresponding concept of row r is one element of the concept set. When we apply a minhash function $f(\cdot)$ to the 0-1 vector of a concept set, it applies a permuation over the rows of it. The minhash value of any vector (concept set) is the subscript of the first row in the permuted order in which the value is 1. The most important property of minhash function $f(\cdot)$ is that

$$\Pr[f(p) = f(q)] = \mathrm{JS}(p,q) \tag{6}$$

Minhash function family $\mathcal{F}$ consists of minhash functions $f(\cdot)$ each manipulates concept sets with different permutation.

To ensure acceptable precision and recall, we use the banding technique [4] to "amplify" the LSH family. By randomly choosing $n(n = r * b)$ minhash functions $f_{i,j} \in \mathcal{F}, i = 1,\ldots,r$ and $j = 1,\ldots,b$ and dividing them into b bands each of r functions, we define the collision (i.e., with same LSH signature) of two concept sets:

Definition 5. $\mathrm{Sig}(p) = \mathrm{Sig}(q) \Leftrightarrow \exists i \in 1,\ldots,b$ s.t. $\forall j \in 1,\ldots,r, f_{i,j}(p) = f_{i,j}(q)$

Suppose the $\mathrm{JS}(p,q) = d$, according to Definition 5, the probability of their collision is $1-(1-d^r)^b$ which is a sharp S-curve (see Figure 2 for example where $r = 8, b = 16$). We should set r, b appropriate values such that $\Pr[\mathrm{Sig}(p) = \mathrm{Sig}(q)]$ near 1 for (p,q) that satisfies $\mathrm{JS}(p,q) \geq t$ while those not so similar bid phrases have low collision probabilities because recalled many false candidates will increase the computational effort (exactly computing the similarity) which means depression of efficiency.

We pre-compute LSH signatures for $p \in P$.For each band, we firstly aggregate bid phrases that have identical signature. Then we maintain the index by a key-value pair memory object caching system where signatures are used as keys and lists of aggregated bid phrases are used as values. We maintain the index of each band in one specific machine. At runtime, when a query is posed, we compute its concept set's LSH signature and distribute signature of each band to corresponding machine. For each band, we check whether the signature of query happens to be the key of a certain key-value pair. If it is the case, value (i.e., list of bid phrases) of the pair is returned because these bid phrases have large Jaccard similarity with the query under some probabilistic guarantee. Finally, we select at most hundreds of bid phrases from results returned from different bands and compute exact Jaccard Similarity for them to select true candidates. By the LSH scheme, we replace cumbersome calculation such as merging hundreds of posting lists of inverted index by accessing b hash tables. Besides, the scalability of our approach partly relies on the banding technique because the disjunctive relationship between different bands enables us to maintain each band in one machine and process in a parallel way. It is this feature that improves the scalability and efficiency of our system.

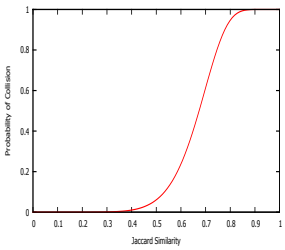


Figure 2: $\Pr[\mathrm{Sig}(p) = \mathrm{Sig}(q)] = 1-(1-d^r)^b$

6.3 Ranking

We assign a semantic-matching score (SMS) to each candidate bid phrase with respect to a given query. Then we rank these candidates by their assigned SMS score in descending order and select top ranking bid phrases. Intuitively, SMS should be proportional to the semantic similarity between short texts. Since each short text is characterized by a concept vector, we can estimate the similarity between short texts by measuring the similarity between their corresponding concept vectors. Formally, for a given query q and a bid phrase p, with their corresponding concept vectors denoted as $\mathbf{c_q}$ and $\mathbf{c_p}$, SMS is defined as follows:

Definition 6.

$$\mathrm{SMS}(q,p) = \cos(\mathbf{c_q}, \mathbf{c_p}) = \mathbf{c_q} \cdot \mathbf{c_p} \tag{7}$$

7. EXPERIMENT

This section describes the experiment settings and analyses the experiment results. Experiments described in this section mainly made use of the following data:

- **Keyword set:** a massive dataset consisting of 0.702 billion bid phrases.
- **Query set:** a collection of 30 million search queries.
- **Search logs:** 15.5 billions of records of the form: <query:str, bid phrase:str, ..., click-or-not:bool> within which there exist 0.83 billion distinct queries.

instance	#senses	#correct	#sample	accuracy
apple	3	157	171	0.918
blackberry	2	127	142	0.894
palm	3	76	84	0.905
bass	4	46	57	0.807
mouse	2	184	200	0.920

Table 3: Accuracy of disambiguation

All these data are provided by a commercial search engine and the search log is accumulated during one month period of time.

7.1 Disambiguation

With rich isA relationships, knowledge-based approaches can map terms to their synonyms and similar varieties through their hypernyms. However, existing approaches neglected polysemy which is also prevalent especially within bid phrases. We have emphasized that leveraging our co-occurrence network, we can identify the appropriate sense of detected instnace, so we made experiment to evaluate the effect of our disambiguation approach.

We randomly sampled one million bid phrases from our keyword set and then extracted those containing ambiguous instances "apple", "blackberry", "palm", "bass" and "mouse" as our test set. We applied our approach to conceptualize each of these bid phrases. Thus each occurrence of the five ambiguous instances is explicitely associated with one of its sense. Regarding the disambiguation as a classification problem, we manually labeled correct sense for these ambiguous instances appeared in test set and adopted accuracy as metrics to access the effect of our disambiguation approach.

The experiment results are presented in Table 3. The classification accuracy of "bass" seems not satisfiable because multi-class classification is much more tough in most cases. Most of the miss classified samples of "apple" are those which refer to "tree" but associated with "fruit" and vice versa. The reason is that seperating these two senses from "company" is easy but the two senses themselves are somehow similar. It is more difficult to identify the proper sense under a smaller granularity.

7.2 CSCD v.s. CSJS

Objective of this experiment is to compare the two candidates selection approaches proposed in this paper. The CSCD (see Subsection 6.1) mines click-through data to extract co-click relationships between queries. Although the co-click relationships capture semantic relatedness between queries accurately, there are inadequate click-through data for tail queries. Thus, CSCD can't be applied for tail queries. On the other hand, CSJS (see Subsection 6.2) directly leverages semantic knowledge provided by Probase and thus has broader application scenarios than CSCD.

Dataset used for this experiment includes our keyword set as well as 21 queries randomly sampled from our search logs. These queries are all related to a specific domain—"hotel". Because "hotel" is one of the three most popular topics(i.e., "travel", "hotel", "insurance") of sponsored search, there is sufficient click information for these sampled queries, and thus CSCD can perform well for them.

For each sampled query, we ranked candidate bid phrases that are selected by CSCD and CSJS respectively based on SMS and reserved at most top-30 as retrieval results.

Score (Label)	Reasons
2 point very relevant	Synonymy;Word Permutation; CAL;Disambiguation.
1 point somewhat relevant	Subset;Overlap; Superset;Similar.
0 point Irrelevant	Clarion hotels→ hotels rewards cards

Table 4: Labeling Guideline for Hotel Queries and Bid Keywords

Metrics used for accessing retrieved results is average precision at N ($P@n$) [5]. We manually labeled more than 1200 query-bid phrase pairs according to the labeling guideline listed in Table 4. We computed P@n by regarding bid phrases with score 1 or 2 as good(positive) cases, and average *"extreme"* precision at n by regarding only bid phrases with score 2 as good(positive) cases.

label	CSJS		CSCD	
	#count	ratio	#count	ratio
0	15	0.062	2	0.010
1	174	0.719	90	0.429
2	53	0.219	118	0.562
P@10	93.80%		99.05%	
extreme P@10	21.90%		56.19%	

Table 5: P@10 of CSJS and CSCD

label	CSJS		CSCD	
	#count	ratio	#count	ratio
0	38	0.082	3	0.007
1	332	0.719	196	0.467
2	92	0.199	221	0.526
P@20	91.77%		99.29%	
extreme P@20	19.91%		56.62%	

Table 6: P@20 of CSJS and CSCD

Experiment results are illustrated in Table 5 and Table 6. As you can see, CSCD outperforms CSJS which confirms the intuition that co-click relationships can accurately capture semantic relatedness. However, not all topics are as popular as "hotel". When there is no sufficient click-through data for CSCD to draw reliable conclusions, CSJS will function as CSCD's complement.

7.3 Validating Semantic-Matching Score

We ranked candidate bid phrases by *SMS* and select the top-ranking ones as resulting expansions for the issued query. Obviously, only when SMS reflects the relevance between short texts accurately can we rank bid phrases in a reasonable order.

7.3.1 Ground Truth

As we know, click thorugh rate (CTR) of an ad is defined as the number of clicks on this ad divided by the number of the ad's impressions. High CTR indicates successful advertising campaign. Because the matching of query to ads is through bid phrases, we can apply CTR for bid phrases by

simply accumulating impressions according to bid phrases instead of ads. Accordingly, we define CTR of query-bid phrase pair as follow:

Definition 7. Given query q and bid phrase p, we have

$$\mathrm{CTR}(q,p) = \frac{\mathrm{Clicks}(q,p)}{\mathrm{Impressions}(q,p)} \times 100\% \tag{8}$$

where both clicks and impressions are triggered by bid phrase p in response to query q.

We extracted tuples in the form $< query : string, bidphrase : string, click - or - not : bool >$ from our search logs. Then we computed CTR of each query-bid phrase pair by aggregating all these tuples over their first two columns.

We made assumption that CTR of query-bid phrse pair is proportional to their semantic similarity and used CTR as ground truth. Thus, to check whether SMS properly reveals semantic similarity between short texts, we check whether there exists a strong positive correlation between SMS and CTR. In another word, if higher SMS corresponds to higher CTR and lower SMS corresponds to lower CTR, we have confidence to announce that SMS can reflect semantic similarity between short texts accurately.

7.3.2 SMS v.s. CTR

We quantitate SMS evenly into ten intervals denoted by integral identifiers ranging from 1 to 10 where interval with larger identifier corresponds to higher SMS. To observe the relationship between SMS and CTR, we present Figure 3(a). Horizontal axis of it represents the identifiers of SMS intervals. Vertical axis of it represents the average CTR of query-bid phrase pairs. As you can see, pairs with higher SMS have higher CTR. There exists a strong positive correlation between them which persuasively proves the effectiveness of SMS.

It is well known that the volume distribution of search queries follows the power law [9]. We divide the query volume into ten deciles evenly.

- *Head queries* belong to deciles $1 \sim 3$. They are extremely popular and searched frequently.
- *Torso queries* belong to deciles $4 \sim 6$.
- *Tail queries* belong to deciles $7 \sim 10$. These queries are rarely accessed and thus have limited historical data.

We show the relationships between SMS and CTR of query-bid phrase pair for head, torso and tail queries in Figure 3(b), 3(c), 3(d) respectively. Although SMS fails to reveal the semantic similarity between head queries and bid phrases accurately, both keyword suggestion and query expansion for these popular queries have been resolved well. Consequently, we can leverage co-click signals described in subsection 6.1 for head queries. As you can see, for torso and tail queries, there exists a strong positive correlation between SMS and CTR of query-bid phrase pair, which confirms the effectiveness of SMS. Because these queries lack of user behavior data, they can't be covered by most existing approaches and are research topics of this area.

7.3.3 Ranking by SMS

Objective of this experiment is to check whether the ranking determined by SMS is reasonable. Although the strong correlation between SMS and CTR confirms the effectiveness of SMS, gathering such statistic(i.e., CTR) globally may fail to distinct an approach that ranks highly relevant bid phrases at the top from another approach that ranks mildly relevant bid phrases at the top.

Dataset for this experiment are 2504 queries(1000 head queries, 1000 torso queries and 504 tail queries) randomly sampled from our search logs. Firstly, we computed CTR of query-bid phrase pair for these sampled queries. Then we rank each query's bid phrases by their CTRs in descending order and reserve the top 30 ones (there are only 504 tail queries that contain at least 30 bid phrases). The ranking as well as these bid phrases' CTRs are used as ground truth for this experiment.

Metrics should take the ranking orders of retrieved bid phrases into account so that more reasonable rankings can be distinguished from unreasonable ones. Thus, we choose Normalized Discounted Cumulated Gain(*NDCG* [5]) as our metrics. Given a sampled query q, aggregate records of search logs to compute CTR of bid phrases with respect to q. Then we rank bid phrases associates with q by their CTR and reserve the top 30 bid phrases $(p_1, \ldots, p_{30})$. We compute the Ideal Discounted Cumulated Gain (*IDCG* [5]) of q by

$$\mathrm{IDCG}(q) = \mathrm{CTR}(q,p_1) + \sum_{i=2}^{30} \frac{\mathrm{CTR}(q,p_i)}{\log_2 i} \tag{9}$$

If we rank the top 30 bid phrases of q by some approach, we derive $p_{f(1)}, \ldots, p_{f(30)}$. Then we compute the Discounted Cumulated Gain (*DCG* [5]) by

$$\mathrm{DCG}(q) = \mathrm{CTR}(q,p_{f(1)}) + \sum_{i=2}^{30} \frac{\mathrm{CTR}(q,p_{f(i)})}{\log_2 i} \tag{10}$$

Finally, NDCG of q can be computed by

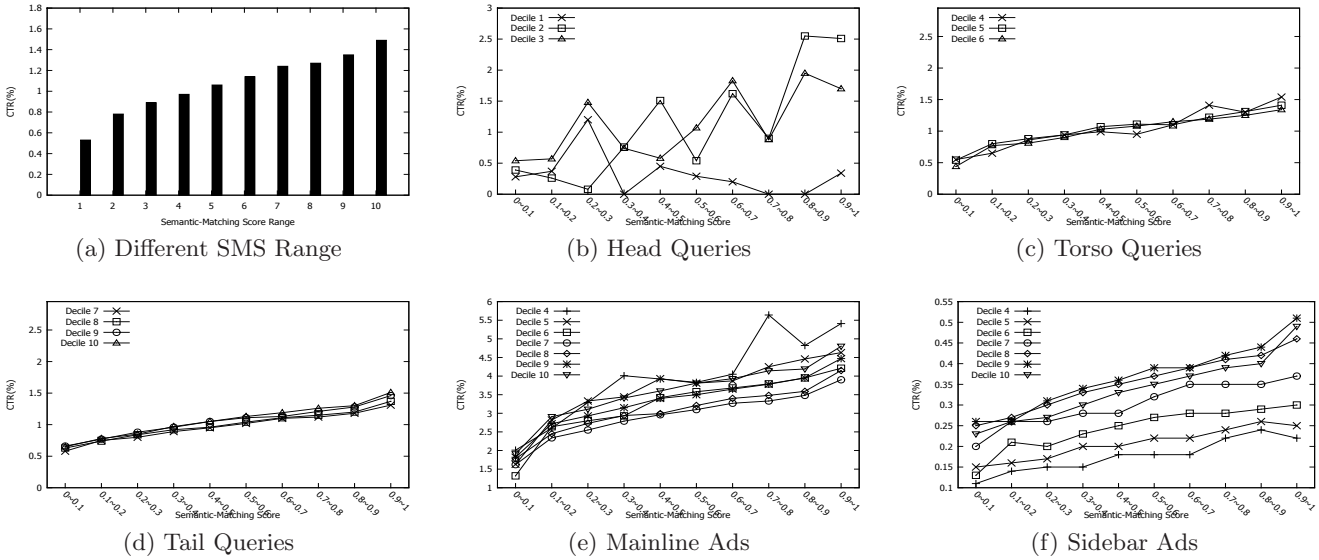
$$\mathrm{NDCG}(q) = \frac{\mathrm{DCG}(q)}{\mathrm{IDCG}(q)} \tag{11}$$

Baselines include bag-of-words technique, pLSA [17] and ESA [15]. We weight terms using classic TF-IDF scheme and set the number of topics of pLSA to 100. TF-IDF weights and probabilities of pLSA model are estimated with respect to all sampled queries as well as their bid phrases.

Experiment results are illustrated in Figure 4. Our approach outperforms baselines under all kinds of settings. Bag-of-words technique can't recall sufficient bid phrases so it loses many bid phrases' "gains" and results in low NDCG. Bid phrases' distributions over latent topics are extremely sparse because each bid phrase contains limited words. Both ESA and our approach model short texts as explicit concepts and thus reveal satisfactory coverage and accuracy. Our approach beats ESA mainly because of the different strategies of conceptualization. ESA maps short text into wikipedia's concept space based on co-occurrence of terms within the short text itself and terms of concepts' corresponding articles. We associate short texts with relevant concepts according to entities of them. Since both queries and bid phrases are very short, conceptualization directly via *isA* relationship is more robust than strategy of ESA.

7.4 Retrieval Evaluation

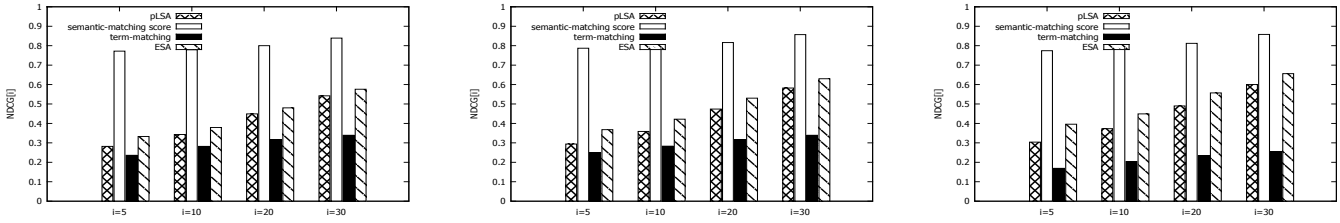
Approach proposed in this paper intends to match a given query to relevant bid phrases. Although previous experi-



(a) Different SMS Range (b) Head Queries (c) Torso Queries

(d) Tail Queries (e) Mainline Ads (f) Sidebar Ads

Figure 3: Correlation between Semantic-Matching Score and CTR



(a) NDCG on head queries (b) NDCG on torso queries (c) NDCG on tail queries

Figure 4: NDCG

ment has confirmed the effectiveness of SMS, ranking a small set of bid phrases in a reasonable order is only a subproblem of retrieving relevant ones from a extremely huge corpus. Evaluation from the standard IR perspective is indispensable.

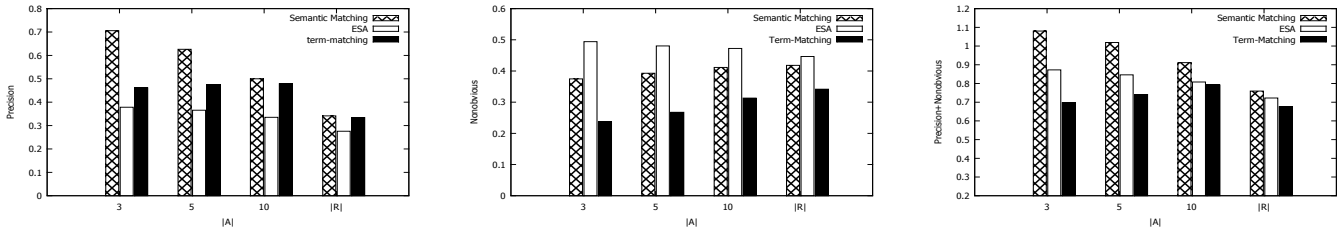
7.4.1 Comparison with Existing Approaches

Dataset for our evaluation purpose includes a set of testing queries and a corpus of bid phrases. We randomly sampled 2554 queries from our search logs as testing queries. We also sampled 300,000 bid phrases from our keyword set which have no common term with our testing queries. We treat these bid phrases as negative (not relevant) cases if they appear in the retrieval results of any test query. Contemporary commercial search engines provide keyword search tools for advertisers to expand a given query with a list of highly relevant bid phrases. We make use of keyword search tool provided by a commercial search engine to expanded our test queries. Among the 2554 test queries, 2025 test queries are matched to at least 5 bid phrases and 1845 queries are matched to at least 10 bid phrases. We regard these bid phrases as positive(relevant) cases if they appear in the retrieval results of their corresponding test queries. We merge the 300,000 sampled bid phrases and those provided by tool together to form our corpus.

Metrics of this experiment includes $p@n$ which reflects the relevance between a given query and its retrieval results, *nonobvious score* which is defined as one minus the Jaccard Similarity between query and bid phrase where both the short texts are regarded as set of terms. In contrast to standard information retrieval which only focuses on the relevance between query and returned documents, keyword suggestion also takes the obviousness of keywords into account because those relevant yet not so obvious bid phrases are not only helpful but also economical. For each test query, its *nonobvious score* is defined to be the average *nonobvious score* of its retrieved bid phrases that are regared as positive cases.

Baselines includes standard bag of words approach as well as ESA [15]. To compare our approach with them, we apply these approaches to selecting bid phrases for each test query from our corpus respectively. Then we evaluate the retrieval results of different approaches with our metrics.

Experiment results are illustrated in Figure 5. Our semantic matching approach outperform ESA and term-vector approach on the metrics $p@n$ which comfirms the effectiveness of our approach from the perspective of information retrieval. Because both ESA and our approach model short text at semantic level while standard bag of words technique simply characterizes each short text as set of terms, the metrics *nonobvious score* is intrinsically inclined to ESA and our approach.



(a) Compare Precision (b) Compare Nonobvious (c) Compare Precision+Nonobvious

Figure 5: Evaluation using Keyword Research Service

$p@n$	Search Engine	Our Approach
$n = 1$	0.778	0.827
$n = 3$	0.729	0.784
$n = 5$	0.723	0.755
$n = 10$		0.717

Table 7: Comparison between Our Approach and Bing Search Engine

7.5 Performance over Massive Dataset

Objective of this experiment is mainly to confirm the scalability of our approach. Besides, since bid phrases provided by keyword search tool is selected by algorithms instead of human beings, We held user study to assess the relevance between queries and retrieved bid phrases.

Dataset for our evaluation purpose includes a set of testing queries and a extremely huge corpus of bid phrases. For the convenience of user study, we select 50 queries from our query set that are easy for participators to interpret. Because previous experiment compare our approach with several baselines over a corpus of ordinary size(300k bid phrases), we adopt the whole keyword set for this experiment.

Metrics of this experiment is p@n. Because our human resource is limited, we give up graded relevance assessments such as NDCG and thus choose p@n which allow only binary assessments. For each test query, we apply different approaches to selecting bid phrases from our massive corpus respectively. Then participators are asked to label each result as relevant or not according to their own subjective judgment. There are totally five participators and all of them are experienced web users.

Baseline of this experiment is not certain keyword suggestion or query expansion approach. Instead, we use each test query's expansions provided by commercial search engine as baseline. The expansions of queries are bid phrases which have the highest *CTR*.

For some test queries, the number of relevant bid phrases provided by the search engine is less than 10, so we have to present the *p@10* only of our approach.

Experiment results are illustrated in Table 7. For small n, our approach achieve obvious improvement compared with the commercial search engine. Because the number of ads displayed for one search is much less than the number of web pages returned to search users, the relevance between top ranked bid phrases and given query is critical. Therefore, the advantage of our approach is significant.

8. CONCLUSION

In this paper, we proposed a novel approach for keyword suggestion and query expansion. We also conducted a series of experiments to evaluate our approach and the experiment results show that our approach can successfully select relevant yet not so obvious bid phrases for a given query no matter the query is popular or rare. Since most existing approaches exploiting historical data can't be applied to tail queries, this fact confirms that directly making use of semantic knowledge is helpful for sponsored search. Besides, our approach outperforms ESA that also model text snippets as explicit concepts, which gives approval to both Probase and conceptualization algorithm proposed in this paper.

9. REFERENCES

[1] Open directory project. `http://www.dmoz.org`.
[2] wikipedia. `http://www.wikipedia.org`.
[3] Wordnet. `http://wordnetweb.princeton.edu/perl/webwn`.
[4] J. D. U. Anand Rajaraman. *Mining of Massive Datasets*. Cambridge University Press, 2011.
[5] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval: The Concepts and Technology Behind Search*. Addison Wesley, 2011.
[6] R. A. Baeza-yates, C. A. Hurtado, and M. Mendoza. *Query Recommendation Using Query Logs in Search Engines*. 2004.
[7] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *JMLR*, 3:993–1022, 2003.
[8] A. Z. Broder, P. Ciccolo, M. Fontoura, E. Gabrilovich, V. Josifovski, and L. Riedel. Search advertising using web relevance feedback. In *CIKM*, pages 1013–1022, 2008.
[9] A. Z. Broder, P. Ciccolo, E. Gabrilovich, V. Josifovski, D. Metzler, L. Riedel, and J. Yuan. Online expansion of rare queries for sponsored search. In *WWW*, pages 511–520, 2009.
[10] A. Z. Broder, M. Fontoura, E. Gabrilovich, A. Joshi, V. Josifovski, and T. Zhang. Robust classification of rare queries using web knowledge. In *SIGIR*, pages 231–238, 2007.
[11] Y. Chen, G. rong Xue, and Y. Yu. Advertising keyword suggestion based on concept hierarchy. In *WSDM*, pages 251–260, 2008.
[12] H. Cui, J.-R. Wen, J.-Y. Nie, and W.-Y. Ma. Probabilistic query expansion using query logs. In *WWW*, pages 325–332, 2002.
[13] S. C. Deerwester, S. T. Dumais, G. W. Furnas, T. K.

Landauer, and R. A. Harshman. Indexing by Latent Semantic Analysis. *JASIST*, 41:391–407, 1990.

[14] A. Fuxman, P. Tsaparas, K. Achan, and R. Agrawal. Using the wisdom of the crowds for keyword generation. In *WWW*, pages 61–70, 2008.

[15] E. Gabrilovich and S. Markovitch. Computing Semantic Relatedness Using Wikipedia-based Explicit Semantic Analysis. In *IJCAI*, pages 1606–1611, 2007.

[16] M. A. Hearst. Automatic acquisition of hyponyms from large text corpora. In *COLING*, pages 539–545, 1992.

[17] T. Hofmann. Probabilistic latent semantic indexing. In *SIGIR*, SIGIR '99, pages 50–57, New York, NY, USA, 1999. ACM.

[18] J. Hu, G. Wang, F. H. Lochovsky, J. tao Sun, and Z. Chen. Understanding user's query intent with wikipedia. In *WWW*, pages 471–480, 2009.

[19] A. Joshi and R. Motwani. Keyword Generation for Search Engine Advertising. In *ICDM*, pages 490–496, 2006.

[20] R. Mihalcea, C. Corley, and C. Strapparava. Corpus-based and Knowledge-based Measures of Text Semantic Similarity. In *AAAI*, 2006.

[21] Y. Song, H. Wang, Z. Wang, H. Li, and W. Chen. Short text conceptualization using a probabilistic knowledgebase. In *IJCAI*, pages 2330–2336, 2011.

[22] F. M. Suchanek, G. Kasneci, and G. Weikum. Yago: a core of semantic knowledge. In *WWW*, pages 697–706, 2007.

[23] H. Wang, Y. Liang, L. Fu, G. rong Xue, and Y. Yu. Efficient query expansion for advertisement search. In *SIGIR*, pages 51–58, 2009.

[24] W. Wu, H. Li, H. Wang, and K. Q. Zhu. Probase: a probabilistic taxonomy for text understanding. In *SIGMOD Conference*, pages 481–492, 2012.

[25] C. Xiao, W. Wang, X. Lin, and J. X. Yu. Efficient similarity joins for near duplicate detection. In *WWW*, pages 131–140, 2008.