Keyword Generation for Search Engine Advertising using Semantic Similarity between Terms

Vibhanshu Abhishek
Fair Isaac Corporation
Bangalore, India
vibhanshuabhishek@fairisaac.com

ABSTRACT

An important problem in search engine advertising is keyword[1] generation. In the past, advertisers have preferred to bid for keywords that tend to have high search volumes and hence are more expensive. An alternate strategy involves bidding for several related but low volume, inexpensive terms that generate the same amount of traffic cumulatively but are much cheaper. This paper seeks to establish a mathematical formulation of this problem and suggests a method for generation of several terms from a seed keyword. This approach uses a web based kernel function to establish semantic similarity between terms. The similarity graph is then traversed using a watershed algorithm to generate keywords that are related but cheaper.

General Terms

Advertising, Internet, Algorithm

Keywords

Keyword Generation, Semantic Similarity, Sponsored Search, Search Engine Optimization

1. INTRODUCTION

Sponsored search or Search Engine Marketing (SEM) is an advertising form on the internet wherein advertisers pay to appear alongside organic search results. The position of the ads is determined by an auction, where the bid by the advertiser is taken into consideration while computing the final position of the advertisement. The ads are served when a user searches for terms on which the advertiser has placed a bid. All the major search engines follow a pay *per click* model, where the advertiser pays only when a user clicks on the ads. These ads tend to be highly targeted and offer a much better return on investment for advertisers as compared to other marketing methods [18]. In addition the large audience it offers has lead to a widespread adoption of search engine marketing. The revenues from search engine marketing exceed billions of dollars and continues to grow steadily [6].

The total number of distinct search terms is estimated to be over a billion [7], though only a fraction of them are used by advertisers. It is also observed that the search volume of queries exhibits a long tailed distribution. This means that a large number of terms with low search volume cumulatively make up a significant share of the total traffic. An advertiser can either bid for a few high volume keywords or select a large number of terms from the tail. The cost of the topmost position is dependent on the term one is bidding for. The bids for the terms vary a lot, from a few cents for an unpopular term to a few dollars for a high volume keyword. The top slot for *massage* costs $5 whereas a bid for *lomilomi massage* costs 20 cents and for *traditional hawaiian massage* costs 5 cents *per click*. Bartz [7] shows that there is a strong correlation between the search volume and the *cost per click* of these terms. Therefore it makes sense to use a large number of cheaply priced terms.

Even though it's beneficial, given the inherent difficulty in guessing a large number of keywords, advertisers tend to bid for a small number of expensive ones. An automated system that generates suggestions based on an initial set of terms addresses this inefficiency and brings down the cost of advertising while keeping the traffic level similar. Some of these terms might be more specific, which might in turn lead to a better conversion rate once the user clicks on the advertisement, thus increasing the revenue of the merchant. Search engine marketing firms and lead generation firms such as Natpal [3] need to generate thousands of keywords for each of their clients. Clearly, it is important to be able to generate these keywords automatically.

This paper mathematically formulates the problem of using many keywords in place of a few.A method is proposed that can be used by an advertiser to generate relevant keywords given his website. An extended dictionary of keywords is constructed by first crawling the webpages in this website and then expanding the set with search results from a search engine. In order to find relevant terms for a query term semantic similarity between terms in this dictionary is established. A kernel based method developed by Shami and Heilman [16] is used to calculate this relevance score. The similarity graph thus generated is traversed by a watershed algorithm that explores the neighborhood and generates novel suggestions for a seed keyword.

2. PROBLEM FORMULATION

Let the profit from a keyword x be defined as:

$$\pi(x) = T(x)(\delta(x)E(x) - c(x)) \quad (1)$$

where *T(x)* is the number of clicks for a particular keyword x, E is the earning from the sale of a product XYZ, δ is the

[1]The term *Keyword* refers to *phrases, terms* and *query term* in general and these terms have been used interchangeably.

WWW2007, May 8–12, 2007, Banff, Canada.
.

probability that a customer will buy the product XYZ when he arrives at the webpage and $c(x)$ is the cost incurred per click for keyword x.

Given a dictionary D of keywords, a corpus C of webpages, a bidding strategy $\Gamma_{bidding}$ and a keyword k, generate a set of suggested keywords $S(k) = \{s_1, s_2, ..., s_t\}$ and their bids $B = \{b_1, b_2, ..., b_t\}$, such that the aggregate profit is maximized,

$$Maximize \qquad \sum_{i=1}^{t} \pi(s_i) \tag{2}$$

the total cost of advertising using these t terms is bounded by the advertising budget, $Budget_{advertising}$,

$$\sum_{i=1}^{t} T(s_i)c(s_i) \leq Budget_{advertising} \tag{3}$$

It is evident from these equations that there is a trade-off between the number of terms that can be used for the advertisement campaign and the total cost as computed in *equation 3*. Relevant keywords are important as their conversions rate will be higher and hence they'll have higher utility as compared to irrelevant keywords.

This approach can be extended to a set of high volume keywords $K = \{k_1, k_2, ..., s_n\}$ such that the final list of suggestions can be a union of the suggestions for the individual terms

$$S = \bigcup_{i=1}^{n} S(k_i) \tag{4}$$

The first step towards solving the aforementioned problem is generation of a large portfolio of keywords that the advertiser can bid on. Several bidding strategies have been proposed [10, 13] and we assume that the strategy $\Gamma_{bidding}$ has been provided to us. Emphasis of this paper is on describing a new technique for generating a large number of keywords that might be relatively cheaper compared to the seed keyword. Generation of the actual bid will be addressed in future work.

3. PREVIOUS WORK

The area of keyword generation is relatively new, though there has been considerable work in the area of query expansion in Information Retrieval (IR). The different techniques for keyword generation can be broadly clubbed under the following headings: query log and advertiser log mining, proximity searches and meta-tag crawlers.

The search engines use query-log based mining tools to generate keyword suggestions. They try to find out co-occurrence relationship between terms and suggest similar starting from an initial keyword. Google's Adword Tool [1] presents past queries that contain the search terms. It also mines advertisers' log to determine keywords they searched for while finalizing a specific keyword. Yahoo's Keyword Selection Tool [4] uses an approach similar to Adwords where it recommends frequent queries by users that contain the keyword. A new method [7] based on collaborative filtering has been proposed by Bartz that uses the relationship between the query terms in the log and the clicked *URL* to suggest new keywords. However, the terms suggested are ones occur frequently in the query logs and there is a high probability that they are expensive.

Most of the third party tools in the market use proximity based methods for keyword generation. They query the search engines for the seed keyword and appends it with words found in its proximity. Though this technique can generate a large number of suggestions it suffers from the same problem faced by the log mining techniques. It cannot produce relevant keywords that do not contain the original term.

Another method used by services like WordTracker [5] is meta-tag spidering. Many high ranked websites include relevant keywords in their meta-tags. The spider queries the search engine using the seed keyword and extracts meta-tags from the top ranked pages which are then presented as suggestions. Some tools also use the Metacrawler search network [2] to get a list of related keywords.

These methods tend to ignore semantic relationship between words. Recent work by Joshi and Motwani [11] presents a concept called TermsNet to overcome this problem. This approach is also able to produce less popular terms that would have been ignored by the methods mentioned above. The authors introduce the notion of directed relevance. Instead of considering the degree of overlap between the characteristic documents of the term, the relevance of a term B to A is measured as the number of times B occurs in the characteristic documents of term A. A directed graph is constructed using this measure of similarity. The outgoing and incoming edges for a term are explored to generate suggestions.

A considerable amount of work has been done in the IR community for query expansion and computation of semantic similarity. Kandola et al. [12] propose two methods for inferring semantic similarity from a corpus. The first one computes word-similarity based on document-similarity and viceversa, giving rise to a system of equations whose equilibrium point is used to obtain a semantic similarity measure. The other technique models semantic relationship using a diffusion process on a graph defined by lexicon and co-occurrence information. An earlier work by Fitzpatrick and Dent [9] measures term similarity using the normalized set overlap of the top 200 documents, though this does not generate a good measure of relevance. Given the large number of documents on the web, this intersection set is almost always empty.

Traditional query expansion techniques [8] augment a user query with additional terms to improve the recall of the retrieved task. Query expansion techniques like the one proposed by Shami and Heilman [16] are typically used to generate a few suggestions per query for the search task. Though keyword generation and query expansion seem to be similar problems, for keyword generation to be successful hundreds and sometimes thousands of keywords must be generated for the method to be effective.

4. WORDY

When an advertiser chooses to advertise using sponsored search, he needs to determine keywords that best describe his merchandise. He can either enumerate all such keywords manually or use a tool to generate them automatically. As mentioned earlier, guessing a large number of keywords is an extremely difficult and time consuming process for a human being. We design a system called *Wordy* that makes the process of keyword search easy and efficient.

Wordy exploits the power of the search engines to gener-

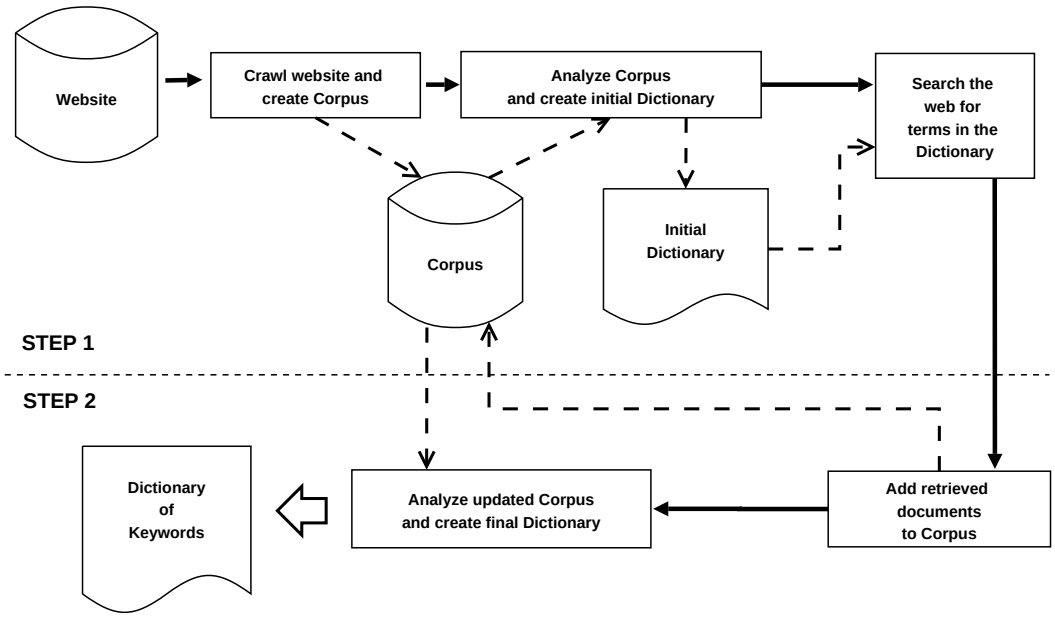

Figure 1: Creation of a large portfolio of keywords.

ate a huge portfolio of terms and to establish the relevance between them. We extend the idea of using search engine for query expansion proposed by Shami and Heilman [16] and apply it to keyword generation. As keyword research needs a lot of suggestions to be effective, their algorithm has been modified so that it is applicable to keyword generation. We adapt their algorithm so that it is better suited to keyword generation. These modifications are described in detail in *Section 5.2*.

We make an assumption that the cost of a keyword is a function of its frequency, i.e., commonly occurring terms are more expensive than infrequent ones. Keeping this assumption in mind a novel watershed algorithm is proposed. This helps in generating keywords that are less frequent than the query keyword and possibly cheaper. The design of Wordy is extremely scalable in nature. A set of new terms or webpages can be added and the system easily establishes links between the existing keywords and the new ones and generates recommendations for the new terms.

5. METHODOLOGY

The task of keyword generation can be broken in three distinct steps, namely

1. Generate a large number of keywords starting from the website of the merchant
2. Establishing sematic similarity between these keywords
3. Suggest a large set of relevant keywords that might be cheaper than the query keyword

This section addresses these steps in detail.

We begin the discussion by defining some terms.

Dictionary D - collection of candidate keywords that the advertiser might choose from.

Corpus C - set of documents from which the dictionary has been generated.

5.1 Initial Keyword Generation

The keyword generation or the dictionary creation process has two steps. This method has been clearly outlined in *Figure 1*. In the first step Wordy scraps the advertisers webpages to figure out the salient terms in the corpus. All the documents existing in the advertisers webpages are crawled and added to the corpus. HTML pages are parsed and preprocessed using an IR package developed at UT Austin [14]. The preprocessing step removes stop words from these documents and stems the terms using Porter's stemmer [15]. After this the documents are analyzed and the *tfidf* of all words in the corpus is computed.

The tfidf vector weighting scheme proposed by Salton and Buckley [17] has been used as it is commonly used in the IR community and is empirically known to give good results. The weight $w_{i,j}$ associated with the term t_i, in document d_i is defined as

$$w_{i,j} = tf_{i,j} \times log(\frac{N}{df_i}), \tag{5}$$

where $tf_{i,j}$ is the frequency of t_i in d_i, N is the total number of documents in the C, and df_i is the total number of documents that contain t_i.

The top d terms in each document weighted by their *tfidfs* are chosen. This set is further reduced by pruning the terms that have a *tfidf* value less than a global *tfidf* threshold, $threshold_{tfidf}$. For terms that occur multiple times the maximum of their ***tfidf*** values is considered. This set of keywords constitute the initial dictionary D_0 as shown in *Step 1* in *Figure 1*. The advertiser can manually add some specific terms like *Anma*[2] to D_0 that might have been elim-

[2] *Anma* is a tradition Japanese Massage

inated in this process. The dictionary thus generated represents an almost obvious set that the advertiser might have manually populated.

In the second step, the dictionary is further expanded by adding terms that are similar to the ones contained in D_0. A search engine is queried for each word in the dictionary. The top l documents are retrieved for each query and they are added to the corpus. All these documents are preprocessed as mentioned earlier in *Step 1* before they are added to the corpus.

$$C = C \bigcup_{i=1}^{|D|} R(w_i) \qquad \forall w_i \in D_0 \tag{6}$$

where $R(w_i)$ represents the documents retrieved from the web for the word w_i. The updated corpus is analyzed and the important terms are determined using the *tfidfs* as mentioned in *Step 1*. These terms are added to the initial dictionary D_0 and the final dictionary D is created. D thus created represents the rich portfolio of terms that the merchant can use for search engine advertising. This process helps the advertiser by finding out a large number of relevant keywords that might otherwise have been missed. An important observation here is that the new terms added to D tend to be more general than the ones that existed in the initial dictionary.

5.2 Semantic Similarity

Once the dictionary D and the corpus C are constructed contextual similarity is established between different keywords in the dictionary. Traditional document similarity measures cannot be applied to terms as they are too short. Techniques like cosine coefficient [17] produce inadequate results. Most of the times the cosine yields a similarity measure of 0 as the given text pair might not contain any common term. Even when common terms exist the returned value might not be an indicator of the semantic similarity between these terms.

We compute semantic similarity between terms in D using a modified version of the technique proposed by Shami and Heilman [16] The authors describe a technique for calculating relevance between snippets of text by leveraging the enormous amount of data available on the web. Each snippet is submitted as a query to the search engine to retrieve representative documents. The returned documents are used to create a context vector for the original snippet, where the context vector contains many terms that occur with the original text. These context vectors are then compared using a dot product to compare the similarity between the two text snippets. Since this approach was proposed to suggest additional queries to the user, it produces a limited set of suggestions for the query term. This method has been adapted here to generate a good measure of semantic similarity between a lot of words which was not the intent of Shamir and Heilman.

This section outlines the algorithm for determining the semantic similarity *K(x,y)* between two keywords x and y.

1. Issue x as a query to a search over the internet.
2. Let *R(x)* be the set of n retrieved documents $d_1, d_2, ..., d_n$
3. Compute the TFIDF term vector v_i for each document $d_i \in R(x)$
4. Truncate each vector v_i to include its m heighest weighted terms
5. Let C be the centroid of the L_2 normalized vector v_i:

$$C = \frac{1}{n} \sum_{i=1}^{n} \frac{v_i}{\|v_i\|_2} \tag{7}$$

6. Let *QE(x)* be the L_2 normalized centroid of C:

$$QE(x) = \frac{C}{\|C\|_2} \tag{8}$$

An important modification made here is that the *tfidf vector* is constructed over *R(x)* for every x. Hence v_i is the representation of document d_i in the space spanned by terms in *R(x)* and not in the space spanned by terms in D. This leads to an interesting result. Lets say there were two words *Shiatsu* and *Swedish Massage* in the dictionary that never occur together in any document. Another word *Anma* appears with *Shiatsu* and *Swedish Massage* separately. When v_i is computed in the manner mentioned above this relationship is captured and similarity is established between the two words *Shiatsu* and *Swedish Massage*[3]. Generalizing, it can be said that $x \sim y$ is established by another term z that does not exist in D.

It has also been discovered that processing the entire document gives better results for keyword generation than processing just the *descriptive text snippet* as mentioned by the authors.

The semantic similarity kernel function k is defined as the inner product of the context vectors for the two snippets. More formally, given two keywords x and y, the semantic similarly between them is defined as:

$$K(x, y) = QE(x).QE(y) \tag{9}$$

The semantic similarity function is used to compute the association matrix between all pairs of terms.

In *Step 4*, the original algorithm truncates the *tfidf vector* to contain only the 50 highest weighted terms. We found that increasing the vector size decreases the number of zero entries in the association matrix, which in turn leads to the discovery of a lot more keywords that are relevant to a given keyword. Currently m is set to *500*, as few documents have more than 500 salient terms. Though there is a decrease in the speed of the system, there is a significant improvement in the number of suggestions generated. Furthermore speed is not such an important factor given the small amount of data we are dealing with as opposed to the enormous amount of query-log data that was processed by Shami and Heilman.

5.3 Keyword Suggestion

The association matrix helps in creating a semantic undirected graph. The nodes of this graph are the keywords and the edges between any two nodes is a function of the semantic similarity between the two nodes.

$$e(x, y) = e(y, x) = 1 - K(x, y) \tag{10}$$

This semantic similarity can be refined using a thesaurus.

For each keyword w_i in the dictionary the number of occurrences in C is computed. It is assumed that frequency

[3]Swedish and Shiatsu are among the massage forms that grew out of Anma

of a word is related to its popularity, terms with higher occurrences would have higher bids. Cheaper keywords can be found by finding out terms that are semantically similar but have lower frequency. A watershed algorithm is run from the keyword k to find such keywords. The search starts from the node representing k and does a breadth first search on all its neighbors such that only nodes that have a lower frequency are visited. The search proceeds till t suggestions have been generated. It is also assumed that similarity has a transitive relationship. $a \sim b \wedge b \sim c \Rightarrow a \sim c$. Suggestions can be generated by substituting as well as appending to the existing keyword k

$watershed\_frequency$:

1. $Queue \leftarrow \{k\}$
2. $S \leftarrow \emptyset$
3. $while((Queue \neq \emptyset) \wedge (|S| < t))$
   (a) $u \leftarrow dequeue(Queue)$
   (b) $S \leftarrow S \bigcup generate\_keywords(S, u)$
   (c) $\forall v \in adj(u)$
      i. $d(v,k) \leftarrow min\{d(v,k), \{e(u,v) + d(u,k)\}\}$
      ii. $if((d(v,k) < thresh) \wedge (freq(v) < freq(u)))$
         A. $enqueue(Queue, v)$
4. $S \leftarrow S - \{k\}$

The user has an option to ignore the preference for cheaper keywords which helps him generate all terms that are similar to the query keyword. This helps him identify popular terms that he might use for his campaign.

6. EXPERIMENTS

The initial corpus consists of 96 documents crawled from websites of 3 spas and 1 dental clinic. The initial dictionary was created by taking top 10 words from each page, out of which 328 were distinct. After further pruning D contained 147 terms. A final dictionary is created by retrieving 10 documents for each word in D_0 using Yahoo Web Services (YWS) API. Finally D contains 1681 terms. For calculating semantic similarity in *Section 5.2*, 25 documents are retrieved to compute the context vector. The representative documents for all terms in D are acquired using YWS.

7. RESULTS

A large number of relevant keyword suggestions can be generated using the technique. For the sake of brevity only the top 10 suggestions generated by *Wordy* have been listed here. Further, concatenation of many of these terms and appending them to the seed keyword can result in more keywords.

skin
skincare
facial
treatment
face
care
occitane
product
exfoliator
dermal
body

teeth
tooth
whitening
dentist
veneer
filling
gums
face
baby
smilesbaltimore
features

pedicure
manicure
leg
feet
nails
treatment
skincare
tool
smilesbaltimore
massage
facial

massage
therapy
bodywork
massageandspalv
therapist
therapeutic
thai
oil
bath
offer
styles

medical
doctor
clinic
health
medicine
service
offers
advice
search
member
information

8. CONCLUSION AND FUTURE WORK

The approach outlined here combines technique from diverse fields and adapts them to solve the problem of keyword generation. The results show that the suggestions generated are extremely relevant and they are quite different from the starting keyword. Wordy is also capable of producing several such suggestions. It has been observed that as the corpus size grows the quality of suggestions improve. Furthermore increasing the number of documents retrieved while creating the dictionary as well as while computing the context vector

increases the relevance of suggested keywords.

Since the proposed solution is heuristic in nature, future work would involve a more rigorous solution to optimize search engine advertising. A metric needs to be developed to measure the efficacy of the system. Currently, only single word terms are considered in this experiment. Extending it to phrases needs no change to the overall framework and is an obvious next step. Integration with systems like WordNet would significantly improve the semantic similarity between these keywords.

9. ACKNOWLEDGMENTS

I would like to thank Dr. Kartik Hosanagar for the many invaluable discussions that signicantly helped this research. I appreciate Yahoo! providing a public API for accessing its webservices.

10. REFERENCES

[1] Google adword https://adwords.google.com/.
[2] Metacrawler http://www.metacrawler.com/.
[3] Natpal http://www.natpal.com/.
[4] Overture http://searchmarketing.yahoo.com/.
[5] Wordtracker http://www.wordtracker.com/.
[6] Iab internet advertising revenue report. Technical report, Price Waterhouse Coopers, April 2005.
[7] K. Bartz, V. Murthi, and S. Sebastian. Logistic regression and collaborative filtering for sponsoreed search term recommendation. In *Second Workshop on Sponsored Search Auctions*, 2006.
[8] C. Buckley, G. Salton, J. Allan, and A. Singhal. Automatic query expansion using smart: Trec 3. *Information Processing and Management*, 1994.
[9] L. Fitzpatrick and M. Dent. Automatic feedback using past queries. In *Proc. of the 20th Annual SIGIR Conference*, 1997.
[10] K. Hosanagar and P. E. Stavrinides. Optimal bidding in search auctions. In *International Symposium of Information Systems*, ISB, Hyderabad, India, 2006.
[11] A. Joshi and R. Motwani. Keyword generation for search engine advertising. In *ICDM'06*, 2006.
[12] J.S.Kandola, J.Shawe-Taylor, and N. Cristianini. Learning semantic similarity. In *NIPS*, 2002.
[13] B. Kitts and B. Leblanc. Optimal bidding on keyword auctions. *Electronic Markets*, 2004.
[14] J. Mooney. Ir package http://www.cs.utexas.edu/users/mooney/cs343/ir.jar.
[15] M. Porter. An algorithm for suffix stripping. *Program*, 1980.
[16] M. Sahami and T. Heilman. A web-based kernel function for matching short text snippets. In *International Conference on Machine Learning*, 2005.
[17] G. Salton and C. Buckley. Term weighting aproaches in automatic text retrieval. *Information Processing and Management*, 1988.
[18] B. K. Szymanski and J.-S. Lee. Impact of roi on bidding and revenue in sponsored search advertisement auctions. In *Second Workshop on Sponsored Search Auctions*, 2006.