Automatic Generation of Bid Phrases for Online Advertising

Sujith Ravi[‡*], Andrei Broder[†], Evgeniy Gabrilovich[†],
Vanja Josifovski[†], Sandeep Pandey[†], Bo Pang[†]

‡ ISI/USC, 4676 Admiralty Way, Suite 1001, Marina Del Rey, CA 90292, USA
† Yahoo! Research, 4301 Great America Parkway, Santa Clara, CA 95054, USA
sravi@isi.edu | {broder | gabr | vanjaj | spandey | bopang}@yahoo-inc.com

ABSTRACT

One of the most prevalent online advertising methods is textual advertising. To produce a textual ad, an advertiser must craft a short *creative* (the text of the ad) linking to a *landing page*, which describes the product or service being promoted. Furthermore, the advertiser must associate the creative to a set of manually chosen *bid phrases* representing those Web search queries that should trigger the ad. For efficiency, given a landing page, the bid phrases are often chosen first, and then for each bid phrase the creative is produced using a template. Nevertheless, an ad campaign (e.g., for a large retailer) might involve thousands of landing pages and tens or hundreds of thousands of bid phrases, hence the entire process is very laborious.

Our study aims towards the automatic construction of online ad campaigns: given a landing page, we propose several algorithmic methods to generate bid phrases suitable for the given input. Such phrases must be both relevant (that is, reflect the content of the page) and well-formed (that is, likely to be used as queries to a Web search engine). To this end, we use a two phase approach. First, *candidate bid phrases* are generated by a number of methods, including a (monolingual) *translation model* capable of generating phrases not contained within the text of the input as well as previously "unseen" phrases. Second, the candidates are ranked in a probabilistic framework using both the translation model, which favors relevant phrases, as well as a *bid phrase language model*, which favors well-formed phrases.

Empirical evaluation based on a real-life corpus of advertiser-created landing pages and associated bid phrases confirms the value of our approach, which successfully re-generates many of the human-crafted bid phrases and performs significantly better than a pure text extraction method.

Categories and Subject Descriptors

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

General Terms

Algorithms, Experimentation

Keywords

Sponsored search, content match

1. INTRODUCTION

In recent years, online advertising has become a $25B+ industry, out of which more than $10B is spent on textual advertising, the ubiquitous short commercial messages displayed alongside Web search results (*sponsored search advertising*) or on third-party Web sites (*content match advertising*). To produce a textual ad, an advertiser must craft a short *creative* (the text of the ad) linking to a *landing page* describing the product or service being promoted. Furthermore the advertiser must associate the creative to a set of manually chosen *bid phrases* representing those Web search queries that should trigger the ad. The same set of bid phrases is indirectly used in content match to decide which ads are suitable for a given page. Since advertisers usually aim to increase their volume, it is desirable for this bid phrase set to be as large as possible; on the other hand, it is also desirable for the bid phrases to be relevant to the product or service being promoted: otherwise, the users are unlikely to click on the ad, or if they click, they are unlikely to *convert*, that is, to purchase the product or the service being offered.[1] For efficiency, given a landing page, the bid phrases are often chosen first, and then for each bid phrase the creative is produced using a template.

Currently, this process is mostly manual, but there are numerous commercial tools to help advertisers choose bid phrases. The majority of these tools, known as "keyword

*The research described herein was conducted while the first author was a summer intern at Yahoo! Research.

[1]For simplicity, we are deliberately ignoring some complexities of sponsored search such as the underlying economics (different keywords have different costs and volumes), advanced matching whereby the search engine display ads on queries that are only related but not identical to bid phrases, relevance/pricing restrictions imposed by the search engines, a more comprehensive definition of conversions, search engine spam, etc. All these issues are not directly pertinent to the main topic of this paper.

WSDM'10, February 4–6, 2010, New York City, New York, USA.

suggestion tools", require an advertiser to enter one or more seed bid phrases, and produce related bid phrases and maybe additional information such as expected volume of queries, costs, etc. Some freely available examples are [9, 12, 21], but there are many more. One problem with these tools is their susceptibility to topic drift: that is, the set often expands towards senses and phrases that have nothing to do with the target. The larger the seed set, the lower the risk of topic drift, but larger seed sets represent more manual work. The technology behind such commercial tools is, of course, a trade secret, but, given the richness and variety of the web queries, the challenge of creating comprehensive bid phrase sets is obvious and has attracted significant scientific attention that resulted in a plethora of approaches for seed set expansion[1, 7, 33, 6, 11, 15].

Since an ad campaign (e.g., for a large retailer) might involve tens of thousands of landing pages, the manual production of even a small seed set is quite laborious and led to the recent appearance of commercial tools that create a bid-phrase sets directly from the landing page [12, 13]. The main aim of this paper is exactly this problem: we present and analyze a bid phrase generation method directly from landing pages, and propose an evaluation technique that uses as a reference the best efforts of actual advertisers. Note that the terms of use of free commercial tools generally preclude bulk uploads and thus we cannot directly compare our method against their technology. But since we compare our efforts against the actual bid-phrases produced by advertisers that are free to use any manual or automatic method available to them, including all the commercial tools above, we can gain a good intuition of how well various methods perform.

Starting from the landing page is challenging, since no matter how long the promoted product description is, it is unlikely to include many synonymous phrases, let alone other perfectly relevant but rarer queries. Therefore, extractive methods that only consider the words and phrases explicitly mentioned in the given description are inherently limited. Indeed, having analyzed a large real-life corpus of ads, we found that as many as 96% of the ads had at least one associated bid phrase not present in their landing page.

This is a classical problem of vocabulary mismatch between the language of the page and the language of the bid phrases (cf. [30]). Initially, we planned to reduce this problem to a known one, namely, that of content match advertising. In that scenario, relevant ads are selected for a given Web page (typically a news page, or a blog page) based on their associated bid phrases and creatives. Consequently, we can treat our landing page (typically, a product description) as if it were a Web page on which ads are to be displayed, then match relevant ads for it from a large existing corpus of ads, and finally use the bid phrases of the top-scoring ads to produce bid phrases to be associated to our landing page (possibly re-ranking these phrases, if needed).

Clearly, this approach does not construct any new bid phrases and only "recycles" existing phrases. Interestingly, we also found this approach to perform roughly on par with the baseline system that used a simple extractive method and used cosine similarity with the landing page text to score the candidates. On further thought, this is not so surprising: the context match system (called CMS in the sequel) that we used was a research variant of Yahoo!'s commercial system, which is finely tuned for pages that normally *host* advertising and not for pages that are the *target* of advertising. Consequently, some existing techniques for placing content match ads (such as [4]) might not be appropriate in this setting. We found that CMS only outperformed the baseline system at high recall levels, where the extractive nature of the baseline became a major limitation.

To improve on the CMS method, we use a very different methodology, based on a two phase approach. First, *candidate bid phrases* are generated by a number of methods, including a (mono-lingual) *translation model* capable of generating phrases not contained within the text of the input as well as previously "unseen" phrases. Second, the candidates are ranked in a probabilistic framework using both the translation model, which favors relevant phrases, as well as a *bid phrase language model*, which favors well-formed phrases. We train the translation model on a (mono-lingual) parallel corpus of bid phrases and landing pages, while the language model is trained on a large-scale Web search query log.

The main challenge of experimenting with large real-life datasets is evaluation, since obtaining human judgments is very expensive. We circumvent this issue by using a large collection of existing ads and their landing pages. Every bid phrase associated to a given ad becomes a "labeled" instance of the form (landing page, bid phrase). We then evaluate our methods based on how well they can predict these bid phrases given the landing page.

The contributions of this paper are threefold. First, we propose a data-driven method for automatically generating bid phrases for online ad campaigns. We believe our method is a first step towards constructing entire ad campaigns automatically, without expensive manual labor. Second, we propose a novel application of existing machine translation methods to the problem of bid phrase generation, and show how to combine translation models with language models to produce highly relevant well-formed phrases. Third, we propose an evaluation framework that does not require human judgments, yet provides a rigorous and extensive approach for testing bid phrase generation methods.

2. METHOD

Let l represent a Web page that can potentially be used as a landing page for an ad. Our task is to automatically generate one or more bid phrases b for l. We treat both b and l as bags of words, that is, $b = \{b_1, b_2, ..., b_n\}$, $l = \{l_1, l_2, ..., l_m\}$, where b_i and l_j denote words in b and l, respectively. Our goal in this generation task is to achieve: (a) high precision, so that only highly relevant phrases are generated in order to target the ad to interested users only, and (b) high recall, to ensure that the ad can reach as many interested users as possible. Clearly, simultaneous optimization of both these goals offers a natural tradeoff, which makes the problem even more challenging.

Note that our problem setup differs from keyword generation [34] where the task is to make a binary decision for each word or phrase in the given page, namely, whether it is a good bid phrase or not. One key difference in our setting is that phrase b may not exist as a phrase in landing page l; in fact, not all words in b have to come from l. Our task also resembles a text summarization system in that a short piece of text is generated from a longer piece and the generated text needs to be truthful to the page. On the other hand, a bid phrase is much shorter than a typical summary, and it is perfectly reasonable to generate multiple bid phrases

at the same time. In addition, a bid phrase does not have to be a grammatically correct sentence; instead, it needs to resemble a valid search query that could be submitted by a Web user.

These characteristics motivated our noisy-channel-based approach [31]. It consists of two main components: generating candidate phrases from page l, and then ranking them. We defer discussions of candidate phrase generation to Section 2.4, and start by describing how we rank the given candidate phrases.

2.1 Ranking candidate phrases

In order to rank candidate phrases b for a landing page l, consider the following generative process. Suppose an advertiser intends to generate a "*target*" landing page l for a given intent characterized by a "*source*" bid phrase b. Let $\Pr(l|b)$ denote the probability of generating l from b.

We aim to rank the candidate phrases based on probability $\Pr(b|l)$, i.e., the likelihood of a bid phrase given the landing page l. Clearly, the more relevant phrase b is to page l, the higher $\Pr(b|l)$ is, and vice versa. However, modeling $\Pr(b|l)$ directly is rather difficult since some of the probability mass would be wasted on ill-formed phrases that are unlikely to be chosen as a bid phrase (e.g., "car on at"), hence we apply Bayes' law to rewrite $\Pr(b|l)$ as:

$$\Pr(b|l) = \frac{\Pr(l|b)\Pr(b)}{\Pr(l)}$$

Now we model the likelihood of bid phrase generation via these two independent components $\Pr(l|b)$ and $\Pr(b)$:

- $\Pr(l|b)$: this is called the translation model (TM) in statistical machine translation (SMT) literature [5], since it gives the translation probabilities from bid phrases to landing pages, learned using a *parallel corpus* (see Section 2.2.1 for more details).
- $\Pr(b)$: this is called the bid phrase language model (LM) since it characterizes whether a phrase is likely to be a valid bid phrase. Hence, phrases like "car rental" will be preferred over phrases like "car on at". This distribution can be estimated on a corpus of phrases alone, which is usually easier to obtain in large quantities than a parallel corpus needed to learn the translation model.

Next, we examine how to estimate both $\Pr(l|b)$ and $\Pr(b)$. In particular, we show how various features previously considered useful for keyword extraction can be incorporated in this generative framework in a principled way.

2.2 Translation models

The main goal of the translation model is to bridge the vocabulary mismatch, so that we can give credit to words in a bid phrase that are relevant to the landing page but do not appear as part of it. In this section, we first discuss how to construct a *parallel corpus* from a given set of $b \rightarrow l$ pairs; we then discuss how to learn a translation model from this corpus to estimate $\Pr(l|b)$ for unseen (b, l) pairs.

2.2.1 Construction of the parallel corpus

Suppose we are given a training corpus L_{train} with a collection of landing pages along with bid phrases provided by advertisers (see Section 4.1 for more details). For each landing page l and each bid phrase b associated with it, we generate a parallel training instance from this (bid phrase, landing page) pair as:

$$b_1\ b_2\ ...\ b_n \rightarrow l_1\ l_2\ ...\ l_m$$

For example, a landing page discussing topics related to the actress *Kirsten Dunst* may contain the words *(kirsten, dunst, film, gossip, maxim, girl, ...)*, and the advertiser-generated bid phrases may include *"kirsten dunst movie", "kirsten dunst interview", "kirsten dunst story", ...*, etc. Each of these bid phrases is paired with the landing page words to create a new parallel training instance for the translation model as follows:

kirsten dunst movie → *kirsten dunst film gossip ...*
kirsten dunst interview → *kirsten dunst film gossip ...*
kirsten dunst story → *kirsten dunst film gossip ...*
...

To provide more "bridges" over the vocabulary gap, we construct additional training instances in the following manner: for a given landing page, we process and use the content within the ad associated with the given landing page (i.e., ad creative, ad title, etc.) and create new (bid phrase, ad word content) pairs as parallel training instances.

2.2.2 Estimating the translation model

For each $b_i \in b$, let $t(l_j|b_i)$ be the probability that l_j is generated from b_i. We estimate $\Pr(l|b)$ as

$$\Pr(l|b) \propto \prod_j \sum_i t(l_j|b_i)$$

Following the machine translation (MT) literature, we refer to t as the translation table, which characterizes the likelihood of a token in a landing page being generated from a token in a bid phrase. Once we have a full table for all words in the bid phrase vocabulary and all words in the landing page vocabulary (note that empirically many pairs may have zero translation probability), we can estimate $\Pr(l|b)$ for (phrase, page) pairs we have not seen in the training data. We now examine how to learn the translation table from the parallel training examples. We outline the intuition behind the expectation maximization (EM) algorithm used in MT literature for this purpose. We then examine how we adopt a standard MT model, in particular, the one that is known as *IBM Model 1*, for our setting.

Recall the *kirsten dunst* example given above. Suppose we knew that the word *film* in the landing page is generated by the word *movie* in the bid phrase; or, following the terminology in machine translation, the word *film* should be *aligned* to the word *movie*. If we are given the alignment information for all occurrences of *movie* in the parallel corpus, estimating $t(\textit{film} \mid \textit{movie})$ is trivial: we count the number of times *movie* is aligned to *film*, and the number of times *movie* is aligned to any word in landing pages, and compute the fraction between this two.

In reality, we do not have the word-level alignment given in our training data. If we can estimate the probability of different alignment assignments, then we can compute the expected values of $t(l_j \mid b_i)$ instead. The most native way to do this is to assume each word in b has equal probability of being aligned to all words in the landing page paired

with it in the parallel corpus. A translation model computed based on this alignment would be similar to what we get from straightforward co-occurrence analysis. In the MT literature, the translation table is learned through the expectation maximization (EM) algorithm where the word-level alignments are treated as hidden variables. Both t values and alignments can be initialized with uniform distributions, and are iteratively refined via EM. Since we treat both b and l as bags of words, ignoring the ordering information, we adopted the model known as IBM Model 1 [5], which is the most appropriate model for our setting.

Null tokens.

One technical detail widely adopted in translation models that is important to our setting is the introduction of *null tokens*. Essentially null tokens are introduced to account for words that do not align well with any words on the other side. In our case, adding null tokens to bid phrases is particularly useful as they account for those l_j that are not closely related to b, so that $t(\cdot \mid b_i)$ does not need to waste its probability mass on irrelevant l_j.

We illustrate this point with the following example. Consider a toy parallel corpus:

Honda → Best car dealer *mp3 player → Buy a new ipod*
Honda → Buy a new car *mp3 player → Best price on nano*

Using the EM algorithm for IBM Model 1, we can learn the following translation table:

b_i	**translations**			
	l_j	$t(l_j \mid b_i)$	l_j	$t(l_j \mid b_i)$
mp3	*on*	0.18	*price*	0.18
	nano	0.18	*a*	0.07
	ipod	0.18	*Best*	0.07
	price	0.18	*new*	0.07
Honda	*car*	0.49	*a*	0.07
	dealer	0.25	*Best*	0.07
	a	0.07	*new*	0.07
null	*a*	0.24	*car*	0.03
	Best	0.24	*dealer*	0.01
	new	0.24	*nano*	0.00
	Buy	0.24	*ipod*	0.00
	car	0.03	*on*	0.00

Note that most of the uninformative words in the landing pages ("uninformative" for this toy example) are mostly accounted by the null token; so that the translation table for real words can concentrate on the more interesting words like *car* or *ipod*. If we estimated the translation probabilities as conditional probabilities computed directly from the co-occurrence information, that is, $\Pr(l_i|b_j) = \frac{count(l_j,b_i)}{count(b_i)}$, we would have $\Pr(Best|Honda) = \frac{1}{2}\Pr(car|Honda)$. In contrast, in the translation table computed through EM, $t(car|Honda)$ is much bigger than $t(Best|Honda)$. Thus, the introduction of null tokens, together with the iterative refinement of the parameters through EM, produce a more reasonable translation table than what we could have gotten directly through simple co-occurrence analysis.

Empirical considerations.

Some words in a landing page are more important than others. For instance, given an HTML page, words that appear in titles, headings, etc. are usually more salient features of the page. To emphasize the importance of such words (in both learning and prediction phases), we associate a weight w_j for all $l_j \in l$. For instance, we can assign a low weight for all normal content words, and a higher weight for words with salient HTML tags (Section 4.1 describes how these weights are computed). We then modify the formula for $\Pr(l \mid b)$ as follows:

$$\Pr(l|b) \propto \prod_j (\sum_i t(l_j|b_i))^{w_j}$$

This effectively implies that important words in a landing page (i.e., words with high w_j scores) account for more of the translation probability mass.

Another feature where our setting departs from typical MT setting is the difference in sizes between the bid phrases and the landing pages: landing pages are usually much longer. Each bid phrase will need to be aligned with multiple words in landing pages, thus diluting the probability mass of $t(\cdot \mid b_i)$. Even in the presence of the null token, one null token is unlikely to account for all irrelevant words on the page since the distribution of $t(\cdot \mid null)$ will be very thinly spread out. One possibility would be to insert multiple null tokens into the bid phrase. We take the alternative route that could potentially lead to more noise-reduction, that is, reducing the set of l_j to top n tokens with highest w_j weight scores in the page. This allows the alignment to be focused on the more important words in the page. In addition, we remove from consideration all stop words occurring on the landing page.

We use the GIZA++ toolkit [25] to perform training. We run only 5 iterations of IBM Model 1 to avoid over-fitting. As output, we obtain a translation table containing triplet entries $< b_i, l_j, t(l_j|b_i) >$.

2.3 Bidphrase language model

Advertisers usually tend to choose bidphrases that match popular queries, in order to increase the chances of their ads being shown and clicked. Search query logs are therefore good sources of well-formed bid phrases. In particular, we instantiate $\Pr(b)$ with an n-gram language model.

Since Web queries are typically short, a bigram model will capture most of the useful co-occurrence information. We build a bigram model smoothed by a unigram model, that is, we *backoff* to unigram model so that bigrams not observed in the training data do not get zero probability. The language model is estimated on a large query corpus Q containing roughly 76 million queries from Yahoo! Web search log.

More specifically,

$$\Pr(b) = \prod_i \Pr(b_i|b_{i-1}),$$

where

$$\Pr(b_i|b_{i-1}) = \lambda_1 f(b_i) + \lambda_2 f(b_i|b_{i-1})$$

with $\lambda_1 + \lambda_2 = 1$. Let $c(b_i, b_j)$ be the number of times $b_i b_j$ appear as a sequence in Q, we have

$$f(b_i|b_{i-1}) = \frac{c(b_{i-1}, b_i)}{\sum_j c(b_{i-1}, b_j)}$$

Let $c(b_i)$ be the number of times b_i appears in Q, and $|V|$ be the vocabulary size of Q (i.e., the number of unique tokens in Q), then $f(b_i)$ can be estimated with add-one smoothing:

$$f(b_i) = \frac{c(b_i) + 1}{\sum_j c(b_j) + |V|}$$

Thus, our bidphrase language model prefers phrases with tokens that are likely to appear in queries, as well as those containing pairs of tokens likely to co-occur in the query log. Since word order is not considered particularly important in bid phrases, we could have adapted the bigram model so that it is order insensitive. However, given the size of Q, if $b_i b_j$ occur more often than $b_j b_i$, we preserve this order preference in our model, even though word ordering is ignored in our evaluation measures.

2.4 Generating candidate phrases

In theory, candidate phrases can be all possible phrases in a reasonably large-scale query log Q. Practically, however, to rank all $b \in Q$ according to $\Pr(l \mid b) \Pr(b)$ is not feasible. One possibility is to consider only phrases from the landing page. Indeed, this is one of the candidate generation strategies we consider (see Section 4.4 for more details). However, as we observed earlier, advertisers often want to pick bid phrases that do not appear as a phrase on the landing page. So in addition to the candidates generated from the landing page (which we refer to as the candidate set B_{LP}), we consider an alternate phrase generation strategy using the translation models described in Section 2.2.

First, we pick the n_p most salient words from the page by selecting those with highest w_j scores, where w_j incorporate the HTML-weights described above. For each word, we then pick the n_t most likely translations from its translation table. These "translated" words are then combined in all possible permutations to form the candidate phrases (we refer to this candidate set as $B_{TM_{gen}}$). We also generate shorter n-gram sequences from these phrases and add them to the candidate pool as well. In our experiments, we set $n_p = 3, n_t = 3$. This is effectively a pre-screening to narrow down the entire universe of possible bid phrases to a few that are likely to be relevant to the target page.

3. ALTERNATIVE METHODS

In the previous section we presented a generative model based approach that relies on translation and language models to generate bid phrases. In this section we discuss some other representative approaches that can potentially be employed for bid phrase generation.

3.1 Using content match systems for bid phrase generation

Recall that the goal of a content match system (CMS) is to find the most relevant ads for a given landing page. Typically, CMS performs this task by converting the landing page into a weighted feature vector, where a feature can correspond to a word or phrase on the landing page and its weight denotes the significance of the feature. Similarly, ads are converted into feature vectors. Then, the relevance of a landing page and an ad is computed by applying a similarity function (e.g., cosine, Jaccard) on their corresponding feature vectors. In our experiments, we used a research variant of the Yahoo! commercial content match system.

We now describe how we can generate bid phrases for a new landing page, using such a content match system and an existing corpus of ads with bid phrases. First, we apply CMS to the landing page to obtain a few top scoring ads, while CMS ranks the ads based on their bid values and relevance to the landing page. For each selected ad, among all its bid phrases CMS also chooses the most "appropriate" bid phrase[2]. We pool together the bid phrases of all selected ads and this gives us the *candidate bid phrases* for this approach, denoted by B_{CMS}.

Next we rank the bid phrases present in this candidate set. We tried two different ranking methods:

CMS-induced ranking: Rank the bid phrases in the order they occur in the ranked list computed by CMS. As explained above, CMS ranking is done by taking both bid values and relevance scores into account, and is thus not optimized for our task.

Frequency-based ranking: Rank the phrases based on their number of occurrences in the candidate set.

From our comparison of the two ranking methods, we found that CMS-induced ranking performs better (though the frequency-based ranking method was not too far off). Hence, we only use this ranking approach for the rest of the experiments (described in Section 4).

3.2 Extraction-based system

We also implemented an approach that is extractive in nature and follows the idea of keyword extraction. We generate both words and phrases as candidates for this approach. Similarly to the previous two approaches, this approach works in two phases where bid phrase candidates are generated in the first phase and ranked in the second phrase. In particular, we first pre-process and tokenize the landing page and the landing URL, and extract words and phrases (n-gram word sequences of length <= 5) from the page. We add these as candidates to the candidate pool.

We compute a weight $w_k' = \frac{f_k}{log(N_d)}$ for each word b_k from the candidate bid phrase, where f_k represents the frequency of the word within the candidate phrase, and N_d is the number of documents on the Web that contain the word. This is a variant of TFIDF weighting. We represent each candidate phrase as a weight vector. In addition, we filter out some of the bad candidates by using a threshold on the weight of words that can appear in a candidate phrase. Next, we represent the landing page in the same way as a weight vector (described in Section 4.1) and compute the cosine similarity between the landing page vector and the candidate vector. The similarity scores are used to induce a ranking on the candidate bid phrase set for a given landing page.

3.3 Discriminative system

We also implemented a machine-learned discriminative system, to compare it with our generative model. Note that unlike all the previous systems, which have both candidate generation and ranking steps, the discriminative system performs ranking only; the phrase candidates are supplied to it using any of the other candidate generation approaches.

Given a bid phrase candidate for a landing page, the discriminative system computes various features to do the ranking. Our feature set includes word overlap score and cosine similarity score of the candidate bid phrase with the landing page. Also, we take into account the position of a word from the bid phrase on the landing page. These are represented as binary features, e.g., whether any word in the phrase candidate is present in the title of the landing page, or in its body, and so on.

We use a corpus of landing pages along with bid phrases for training, and train a ranking model using SVM^{rank} [14]

[2]This selection is also based on the bid value and relevance of the bid phrase to the landing page.

with a linear kernel. Then, given a test landing page and set of candidate bid phrases (along with their features), we use the trained model to rank the bid phrases with respect to the given page.

4. EXPERIMENTS

In this section, we first describe the datasets and the evaluation measures. We then present our main evaluation results, followed by an analysis of the effectiveness of various components in our model.

4.1 Data

In order to collect data for our experiments, we sample the Yahoo! ad corpus, and retrieved a set of ads with advertiser-specified bid phrases. Each of these ads is associated with a landing URL, to which users are redirected when they click on the ad. We then crawled these landing URL's to create our data corpus L. To avoid any bias from advertisers with a large number of ads (e.g., www.ebay.com), we randomly sampled 5 landing URL's per domain, which resulted in a set of 54,548 landing pages. For each landing page $l \in L$, we thus had access to the ad associated with it, and also to the set of bid phrases B^* provided by the advertiser for this particular ad. On average, there were 9 bid phrases per landing page in our corpus. Next, we split our landing page corpus L into a *training* set L_{train} (consisting of 40,048 Web pages) and *test* set L_{test} (containing 10,500 Web pages).

Given this training set, we used the approach outlined in Section 2.2.1 to construct a parallel training corpus with approximately 399,000 instances of the form (bid phrase, landing page). Since our data set is very large, we only report results of using a single train/test split but do not perform cross-validation.

Pre-processing the landing pages.

As a pre-processing step, the HTML content of each landing page was first parsed and lower-cased. Additionally, stop words were removed and the content is tokenized. We also applied the same pre-processing steps to the landing URL itself, and added the result to the output of the landing page pre-processing. Next, for each word l_j in the tokenized output, we computed a weight associated with the word, $w_j = \frac{weight_{tag} \times f_j}{log(N_d)}$, where $weight_{tag}$ is a special weight assigned to each word depending on its HTML tag position, f_j is the frequency of the word on the given landing page, and N_d is the number of documents on the Web that contain the word. We set $weight_{tag}$ to a higher value of 10 for words appearing in the landing URL or on the landing page within important tags such as <*title*>, <*keywords*>, <*h1*>, and used $weight_{tag} = 1$ for words appearing elsewhere on the page. Unimportant words occurring on the page were filtered out using a threshold on the weight; in our experiments, we filtered out words with $w_j < 0.5$.

After tokenization, we used the processed L_{train} corpus to train a discriminative ranking model using SVM^{rank} (described in Section 3.3). Also, we used it to construct a parallel corpus to train the translation model (described in Section 2.2.1).

4.2 Evaluation measures

In this section we describe the evaluation procedure. To facilitate large-scale evaluation, we focus on automatic evaluation measures using the "gold standard" bid phrases provided by the advertisers. Since the gold standard phrases may not cover all possible bid phrases that are relevant to the page, such automatic measures might under-value valid phrases. However, we believe that the relative ordering is still meaningful, and focus on the comparisons among different systems, rather than on the raw numbers.

For a given landing page $l \in L_{test}$, we look up the set of gold bid phrases B^*. We then compare bid phrases proposed by various systems against B^*. We apply stopwords removal and stemming (using the Porter stemmer [26]) prior to the comparison.

Since we want to evaluate and compare *phrases* and not *words*, we cannot employ conventional precision/recall measures. If we were to evaluate the bid phrases using the standard notion of precision, shorter bid phrases (and most likely single word candidates) would be preferred and this would defeat our purpose. In fact we prefer high quality focused bid phrases and not simple keywords. Hence, we employ two different measures to evaluate different aspects of the bid phrases generated by our systems. We wish to capture the co-occurrence of words occurring in a bid phrase and not necessarily the sequence in which they appear in the phrase. So the evaluation measures we design are order-insensitive (for example, the phrases *"toyota hybrid"* and *"hybrid toyota"* are considered equivalent and assigned the same score).

Edit distance: The first measure we employ is normalized edit distance [24], which is a well-studied measure commonly used in information retrieval [10, 23] and natural language processing applications [8, 29] to determine the similarity of two strings by computing partial sub-string matches instead of an absolute match. Let, for test landing page l, B^* denote the set of gold bid phrases and b denote the top ranked candidate phrase generated by our approach. We compare candidate phrase b to a gold bid phrase $b^* \in B^*$ using a *word-level edit distance measure*, i.e., the average number of words required to be inserted, deleted or substituted within the phrase b in order to convert it to the gold phrase b^*. In other words,

$$ED(b, b^*) \quad = \quad \frac{\text{\# of operations to convert } b \rightarrow b^*}{\text{\# of words in } b^*}$$

Since any $b^* \in B^*$ is considered a good answer, we compute the edit distance ED for b with respect to every gold bid phrase $b^* \in B^*$ and pick the minimum:

$$minED(b, l) \quad = \quad min_{b^* \in B^*} \; ED(b, b^*)$$

A lower *minED* score implies higher relevance. For example, if the candidate bid phrase, $b =$ *"bmw x5"* and the gold bid phrases are $b^{*(1)} =$ *"2009 bmw x5"* and $b^{*(2)} =$ *"2009 bmw x5 dealers"*, then,

$$ED(b, b^{*(1)}) = \frac{1}{3} = 0.33$$

$$ED(b, b^{*(2)}) = \frac{2}{4} = 0.50$$

$$\text{and, } minED(b, \{b^{*(1)}, b^{*(2)}\}) = 0.33$$

Since advertiser may want multiple bid phrases for a given page, we also evaluate the average quality of a set of top phrases generated by each system. To this end, we generalized the *minED* score as:

$$minED@ \text{ rank N} \quad = \quad \frac{\sum_{r=1 \ldots N} minED(b^{[r]}, l)}{N}$$

where $b^{[r]}$ denotes the r^{th} ranked bid phrase candidate.

ROUGE-1 score: The second evaluation method that we employ is a recall-based measure widely used in natural language processing applications such as document summarization and machine translation [18, 19]. This measure has been well-studied and shown to correlate well with human evaluations [17, 18]. We adapt this measure to our task and use it to evaluate the quality of a candidate bid phrase against all the relevant gold bid phrases and not just the best matching one, unlike *minED*.

$$\textit{ROUGE-1}(b, l) \quad = \quad \frac{\sum_{b^* \in B^*} \#\text{ of words in } b \cap b^*}{\sum_{b^* \in B^*} \#\text{ of words in } b^*}$$

A higher *ROUGE-1* score implies higher relevance. For example, if two of the candidate bid phrases proposed for a given landing page are $b^{(1)}$ = *"engine oil"* and $b^{(2)}$ = *"vw car engine oil"* and the gold bid phrases are *"volkswagen jetta engine oil"*, *"volkswagen engine oil"*, and *"vw engine oil"*, then the *minED* scores for the two candidates are both 0.33. Even though the second bid phrase is a more relevant and informative candidate for the given landing page, the *minED* scores both of the candidates equally. But according to the ROUGE-1, the scores for the two candidates are 0.6 and 0.7 respectively, thereby correctly assigning a higher score for the second candidate. In general, a bid phrase generation system should perform well in terms of both evaluation measure (*minED* and *ROUGE-1*) to generate high quality bid phrases.

We can also compute the average ROUGE-1 score for the top N bid phrases:

$$\textit{ROUGE-1@ rank N} \quad = \quad \frac{\sum_{r=1 \ldots N} \textit{ROUGE-1}(b^{[r]}, l)}{N}$$

4.3 Main Comparisons

Next we compare the performance of the following four bid phrase generation systems (described earlier), each using the candidate set that is naturally available to the method:

- Extraction-based (Baseline): Generates and ranks candidates from the landing page. Details in Section 3.2.
- Content-Match-based (CMS): Details in Section 3.1.
- Translation-based (LM+TM): Details in Section 2.
- Discriminative: Details in Section 3.3. In our experiments, we give it the candidate phrase set of both CMS and baseline system.

Table 1 summarizes the performance using both evaluation measures. Note that the average performance tend to decrease as N becomes larger, because it is more difficult to generate more candidate phrases and still maintain the same overall quality.

In terms of *minED*, the LM+TM system outperforms all other systems in most settings, especially at higher ranks where the margin of improvement is higher. The same trend is observed for *ROUGE-1* evaluation, where translation-based approach (LM+TM) consistently produces better scores than all other systems at all ranks. In fact, our translation-based system achieves an impressive 21-70% improvement over the extraction-based baseline in *ROUGE-1* scores at various ranks. This shows that the translation approach yields good quality bid phrases (achieving low edit distance scores) and at the same time allows us to better capture the diversity in the gold bid phrase set (as shown by higher *ROUGE-1* scores). This improvement can be attributed to: (a) the LM+TM system can generate candidates that are present on the landing page as well as the unseen ones via the translation model, thereby enhancing its candidate pool (Section 2.4) and (b) the ranking is better when both language model and translation model are used, whereby the language model ensures well-formed phrases while the translation model ensures high relevance. We study this in more details in Section 4.4 and 4.5.

Also, we note that the Content Match System (CMS) did not perform as well as the baseline system, especially in terms of edit distance, though it does yield some improvements in *ROUGE-1* scores at higher ranks. We will discuss this in more detail in Section 4.6. The discriminative system which performs feature-based ranking performs reasonably well: achieving comparable *minED* and better *ROUGE-1* scores than the baseline, but it still does not match the LM+TM system.

The performance reported may look on the low side. Apart from the caveats of the automatic evaluation (that gold set might not include all relevant phrases), we are also dealing with a very difficult task. For each landing page, we measured the overlap of its gold bid phrases with the page content. We observed that almost 96% of the landing pages in the test corpus had at least one gold bid phrase that did not appear on the page. Furthermore, the gold bid phrase sets for at least 70% of the test landing pages contained one or more words that were not present on the page. This gives an indication of the difficulty of generating bid phrases.

As described earlier (Section 2), the bid phrase generation process involves two major phases, namely, candidate generation, and candidate ranking. We now examine each phase in more detail.

4.4 Comparison of candidate generation

In this section, we study the effect of candidate generation methods, while fixing the ranking method to be the LM+TM ranking. We obtain the candidate pool C from four different candidate generation strategies as shown in Table 2 (details described in Sections 2 and 4). The system achieves the best results in terms of *ROUGE-1* scores when using a combination of candidate sets from multiple sources (B_{LP+CMS} and $B_{LP+TM_{gen}}$). In particular, we find that the system selects the best quality bid phrases in terms of both evaluation measures (at various ranks) when we use the landing page content to generate candidates *directly* (i.e., select words/phrases occurring on the page) as well as *indirectly* (using the translation model to generate new candidate phrases). In other words, the combined candidate set $B_{LP+TM_{gen}}$ proves to be a good source for selecting high quality bid phrases relevant to the given landing page.

	LM+TM			
	B_{LP}	B_{CMS}	B_{LP+CMS}	$B_{LP+TM_{gen}}$
minED @ rank 1	0.66	0.76	0.70	0.68
minED @ rank 5	0.70	0.79	0.72	0.68
minED @ rank 10	0.73	0.81	0.74	0.70
ROUGE-1 @ rank 1	0.25	0.29	0.29	0.29
ROUGE-1 @ rank 5	0.20	0.25	0.26	0.28
ROUGE-1 @ rank 10	0.17	0.23	0.24	0.27

Table 2: Comparison of different candidate generation strategies.

	Baseline (cosine)	CMS	Discriminative System (SVM^{rank} with features)	LM+TM $B_{LP+TM_{gen}}$
minED @ rank 1	0.66	0.78	0.67	0.68
minED @ rank 5	0.71	0.81	0.72	0.68
minED @ rank 10	0.75	0.83	0.74	0.70
ROUGE-1 @ rank 1	0.24	0.22	0.26	0.29
ROUGE-1 @ rank 5	0.19	0.21	0.24	0.28
ROUGE-1 @ rank 10	0.16	0.20	0.22	0.27

Table 1: Comparison of bid phrases generated by various systems for the 10,500 test landing pages in terms of two different measures—(1) minimum edit distance *(minED)*, and (2) *ROUGE-1* score. The average scores for top N bid phrases are shown.

4.5 Comparison of Ranking Methods

Once we generate a candidate pool for any given landing page, we can use any of the four systems described in Section 4.3 to rank the bid phrases from the pool.

4.5.1 Translation-based ranking versus others

In order to ensure that we perform a fair comparison between all ranking methods, we need to choose the same pool of candidate phrases for all the four methods. Due to some system constraints we could not configure the content match system to take a candidate pool as input, hence we choose the candidate pool that CMS uses, i.e., B_{LP+CMS}, as the common pool and supply it to the other methods. We used each system to rank the bid phrases in this pool and compare the quality of the ranked phrases.

Table 3 shows the effect of the different ranking methods on the quality of the top N bid phrases. While the edit distance scores do not vary significantly for different systems, the *ROUGE-1* scores show significant variation. We note that even under this setting, the translation-based approach (LM+TM) outperforms all other systems achieving more than 10% improvement in *ROUGE-1* scores. Our results suggest that the translation-based approach is not only good at generating new bid phrase candidates, but it can also be used as a ranking model with a given candidate set to select high quality bid phrases.

	B_{LP+CMS}			
	Baseline	CMS	Discr. System	LM+TM
minED @ rank 1	0.68	0.78	0.67	0.71
minED @ rank 5	0.72	0.81	0.72	0.72
minED @ rank 10	0.74	0.83	0.74	0.74
ROUGE-1 @ rank 1	0.27	0.22	0.26	0.29
ROUGE-1 @ rank 5	0.23	0.21	0.24	0.26
ROUGE-1 @ rank 10	0.22	0.20	0.22	0.24

Table 3: Comparison of different ranking methods when using the candidate set B_{LP+CMS}.

Next, we perform a deeper analysis of the translation-based system by varying different components within the system and study their effects on the ranked bid phrases.

4.5.2 Component analysis for translation-based ranking

There are two main components in the translation-based ranking system, namely, a translation model and a language model. We can modify these components and observe the effect in terms of bid phrase quality.

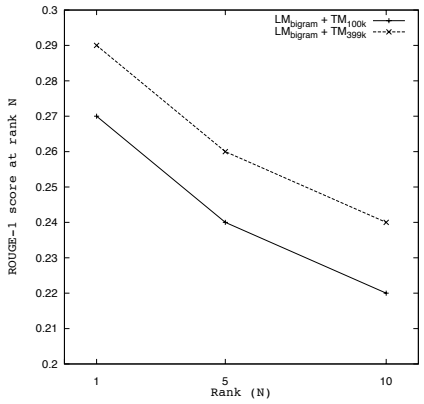

Figure 1: Effect of size of the corpus used to train translation models.

Effect of the translation model in ranking.

Here we study the effect of the translation model used while ranking, by building two separate translation models TM_{100k} and TM_{399k} trained on different data sizes (using 100,000 and 399,000 parallel training examples, respectively). Each of these translation models was then combined with a bigram language model to produce two different ranking systems $LM_{bigram} + TM_{100k}$ and $LM_{bigram} + TM_{399k}$. We used these two ranking systems to score and rank our candidate bid phrases (from the combined set B_{LP+CMS}) and computed the *ROUGE-1* scores at various ranks (edit distance scores showed insignificant variation). Figure 1 demonstrates that as the translation model gets better (i.e., when using more training data), the quality of the bid phrases generated by the system in terms of *ROUGE-1* scores also improves consistently at all ranks.

Effect of the language model in ranking.

We perform an additional experiment where we vary only the language model component that is used during ranking. Figure 2 shows how varying the language model affects the quality of the bid phrases selected by the LM+TM approach. As expected, we found that using a bigram language model (LM_{bigram}) results in improved ranking of the bid phrases (in terms of *ROUGE-1*), as compared to a unigram language model ($LM_{unigram}$).

4.6 Discussion

It is surprising to find that although the CMS system produces novel bid phrases that may not even appear on the page, yet the top bid phrases proposed by the system are

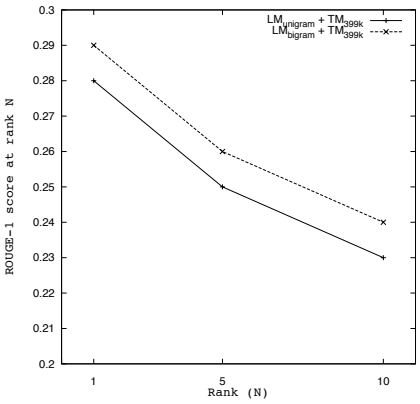

Figure 2: Comparison of unigram vs. bigram language model.

not necessarily of high quality when evaluated against the gold bid phrases. In comparison to the baseline system, CMS performs poorly in terms of edit distance metric (see Table 1). The phrases produced by the CMS system tend to be longer compared to gold bid phrases, so edit distance suffers. But at higher ranks (N = 5, 10), the ROUGE-1 scores for CMS are better than the baseline. For example, at rank 5 the *ROUGE-1* score for CMS system is 0.21 compared to 0.19 achieved by the baseline system. We did a further analysis of the two systems to see whether the CMS system outperforms the baseline in some cases. We observe that the CMS system does perform better than the baseline at all ranks when the word overlap between gold bid phrases and landing page is low.

We note that our translation based approach is superior to all other systems, achieving the best *minED* and *ROUGE-1* scores at various ranks (as shown in Table 1). It serves two purposes, namely, (1) for generating new candidate phrases that do not appear on the page, and (2) for ranking the selected candidates in a probabilistic framework when combined with the language model component. The bid phrases produced by this system are of better quality than other systems (i.e., the bid phrases are well-formed and achieve high scores in terms of both evaluation measures). Using the translation table learned during the training phase, the system is able to generate many relevant bid phrases containing words that might be missing from the page. Our system learns good translations for many words from the bid phrase vocabulary (e.g., *mag* is translated to *mag, magazine, cover, subscription, magazin,* etc.)

5. RELATED WORK

There are two main bodies of prior research that are relevant to our study, namely, those focusing on online advertising and on statistical machine translation.

5.1 Online advertising

Online advertising is an emerging area of research, so the published literature is quite sparse. A recent study [32] confirms the intuition that ads need to be relevant to the user's interest to avoid degrading the user's experience and increase the probability of reaction.

Arguably, the most relevant previous study was conducted by Yih et al. [34], who extracted phrases from the page and matched them to the bid phrases of the ads. The authors described a system for phrase extraction that used a variety of features to determine the importance of page phrases for advertising purposes. The system was trained with pages that have been hand-annotated with important phrases. The learning algorithm took into account features based on TFIDF, HTML meta data, and search query logs to detect the most important phrases. However, the primary aim of this paper was to propose an alternative approach to contextual advertising by reducing it to the problem of sponsored search advertising; this study did not aim at generating bid phrases for ads.

In the content match scenario, Ribeiro-Neto et al. [30] addressed the "impedance mismatch" problem, namely, the discrepancy between the vocabulary used in the ads and in the Web pages. The authors examined a number of strategies for matching pages to ads based on extracted keywords. The authors achieved improved matching precision by expanding the page vocabulary with terms from similar pages, which were weighted based on their overall similarity to the original page. In this paper, we "bridge" between related words by using a mono-lingual translation approach.

Bid phrases are essentially search queries, and hence another relevant research direction is that of query expansion and rewriting [3, 2, 28]. Query rewriting is a common technique for performing broad match in sponsored search, whereas the original query is rewritten into a "better" query, which is more likely to retrieve more relevant ads. On the other hand, query expansion seeks to augment the original query with additional features based on various external sources of knowledge. In this paper, we use the candidate bid phrases "as-is", but in our future work we intend to enhance their representation using techniques developed in the query expansion literature.

5.2 Machine Translation

The noisy channel model is a powerful framework that has been adopted for various natural language processing tasks. Noisy channel approaches have previously been applied to text summarization [16], a task that is similar to our problem in that a shorter piece of text is produced for a longer piece of text; but as discussed in Section 2, our task also has notable differences in comparison to a standard summarization task. In terms of methodology, our approach is directly based on statistical machine translation (SMT) models [20]. The ultimate goal of SMT systems is to translate a sentence in the source (foreign) language to a sentence in the target language. Modern SMT systems are based on noisy channel model, whose parameters are learned from bilingual *parallel* corpora consisting of pairs of sentences in the two languages that are properly *aligned* so that each pair express the same meaning. The first step is to induce an alignment between words on both sides for a given sentence pair. The most widely used models for word alignment are IBM Model 1-4 [5]. The idea of *parallel* corpora has also been extended to monolingual settings, where they contain aligned texts written in the same language, and SMT models are applied to extract paraphrases [27].

SMT has also been used in contextual match for online advertising [22], a different but closely related task. Their final task differs from us in that they concentrate on the task of ranking ads retrieved from the ad database for a given Web page, and do not consider generation of bid phrases.

In their work, IBM Model 1 is used to refine the similarity score between ad and page, which is used as one of the features in their final system. In contrast, we adopted a generative model as the main framework and incorporated various features as organic components in this model.

6. CONCLUSION

We proposed several automatic methods for generating bid phrases for online advertising. Our main approach uses a generative model within a machine translation framework, and the system translates any given landing page into relevant bid phrases. The proposed model uses two components, a language model for selecting well-formed bid phrases and a translation model that helps generate novel bid phrases that might not appear on the page. We also proposed two evaluation measures to assess the quality of the generated bid phrases when compared against the gold standard bid phrases (as specified by the advertiser). Empirical evaluation on a large test corpus of landing pages shows that our translation-based approach outperforms all other systems, including a baseline that uses an extractive approach, in terms of both evaluation measures.

One limitation of our study was in its use of completely automatic evaluation measures, as outlined in Section 4.2. Using such automatic measures allowed us to conduct experiments with a very large real-life dataset. In our future work, we also plan to perform a limited manual evaluation of the quality of bid phrases generated by our method, in order to be able to better characterize its performance.

7. REFERENCES

[1] V. Abhishek and K. Hosanagar. Keyword generation for search engine advertising using semantic similarity between terms. In *EC '07*, pages 89–94, 2007.

[2] A. Broder, P. Ciccolo, M. Fontoura, E. Gabrilovich, V. Josifovski, and L. Riedel. Search advertising using Web relevance feedback. In *Proc. of CIKM*, 2008.

[3] A. Broder, P. Ciccolo, E. Gabrilovich, V. Josifovski, D. Metzler, L. Riedel, and J. Yuan. Online expansion of rare queries for sponsored search. In *Proc. of WWW*, 2009.

[4] A. Broder, M. Fontoura, V. Josifovski, and L. Riedel. A semantic approach to contextual advertising. In *SIGIR'07*, pages 559–566. ACM Press, 2007.

[5] P. Brown, V. Della Pietra, S. Della Pietra, and R. Mercer. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311, 1993.

[6] W. Chang, P. Pantel, A.-M. Popescu, and E. Gabrilovich. Towards intent-driven bidterm suggestion. In *WWW '09*, pages 1093–1094, 2009.

[7] Y. Chen, G.-R. Xue, and Y. Yu. Advertising keyword suggestion based on concept hierarchy. In *WSDM'08*, 2008.

[8] T. Doi, H. Yamamoto, and E. Sumita. Example-based machine translation using efficient sentence retrieval based on edit-distance. *ACM Transactions on Asian Language Information Processing (TALIP)*, 4(4):377–399, 2005.

[9] `freekeywords.wordtracker.com`. Retrieved Aug. 13, 2009.

[10] J. C. French, A. L. Powell, and E. Schulman. Applications of approximate word matching in information retrieval. In *Proc. of CIKM*, 1997.

[11] A. Fuxman, P. Tsaparas, K. Achan, and R. Agrawal. Using the wisdom of the crowds for keyword generation. In *WWW'08*, pages 61–70, 2008.

[12] `adwords.google.com/select/KeywordToolExternal`. Retrieved Aug. 13, 2009.

[13] `www.google.com/sktool`. Retrieved Aug. 13, 2009.

[14] T. Joachims. Training linear svms in linear time. In *Proc. of SIGKDD*, 2006.

[15] A. Joshi and R. Motwani. Keyword generation for search engine advertising. In *ICDMW'06*, pages 490–496, 2006.

[16] K. Knight and D. Marcu. Statistics-based summarization - step one: Sentence compression. In *Proc. of AAAI*, 2000.

[17] C. Lin. ROUGE: A package for automatic evaluation of summaries. In *Proc. of the Workshop on Text Summarization Branches Out, ACL (WAS)*, 2004.

[18] C. Lin and E. Hovy. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proc. of NAACL/HLT*, 2003.

[19] C. Lin and F. J. Och. Automatic evaluation of machine translation quality using longest common subsequence and skip-bigram statistics. In *Proc. of ACL*, 2004.

[20] A. Lopez. Statistical machine translation. *ACM Comput. Surv.*, 40(3):1–49, 2008.

[21] `advertising.microsoft.com/search-advertising/advertising-intelligence/keyword-suggestion`. Retrieved Aug. 13, 2009.

[22] V. Murdock, M. Ciaramita, and V. Plachouras. A noisy-channel approach to contextual advertising. In *ADKDD Workshop, KDD*, 2007.

[23] G. Navarro. Improved approximate pattern matching on hypertext. *Theor. Comput. Sci.*, 237(1-2):455–463, 2000.

[24] G. Navarro. A guided tour to approximate string matching. *ACM Comput. Surv.*, 33(1):31–88, 2001.

[25] F. J. Och and H. Ney. Improved statistical alignment models. In *Proc. of ACL*, 2000.

[26] M. F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

[27] C. Quirk, C. Brockett, and W. Dolan. Monolingual machine translation for paraphrase generation. In *Proc. of EMNLP*, pages 142–149, 2004.

[28] F. Radlinski, A. Broder, P. Ciccolo, E. Gabrilovich, V. Josifovski, and L. Riedel. Optimizing relevance and revenue in ad search: A query substitution approach. In *Proc. of SIGIR*, pages 403–410, 2008.

[29] S. Ravi and K. Knight. Learning phoneme mappings for transliteration without parallel data. In *Proc. of HLT/NAACL*, pages 37–45, 2009.

[30] B. Ribeiro-Neto, M. Cristo, P. B. Golgher, and E. S. de Moura. Impedance coupling in content-targeted advertising. In *Proc. of SIGIR*, 2005.

[31] C. E. Shannon. A mathematical theory of communication. volume 27, pages 379–423, 1948.

[32] C. Wang, P. Zhang, R. Choi, and M. D. Eredita. Understanding consumers attitude toward advertising. In *8th Americas Conference on Information Systems*, pages 1143–1148, 2002.

[33] H. Wu, G. Qiu, X. He, Y. Shi, M. Qu, J. Shen, J. Bu, and C. Chen. Advertising keyword generation using active learning. In *WWW'09*, pages 1095–1096, 2009.

[34] W. Yih, J. Goodman, and V. R. Carvalho. Finding advertising keywords on Web pages. In *Proc. of WWW*, 2006.