# A Semantic Approach to Contextual Advertising

Andrei Broder Marcus Fontoura Vanja Josifovski Lance Riedel
Yahoo! Research, 2821 Mission College Blvd, Santa Clara, CA 95054
{broder, marcusf, vanjaj, riedell}@yahoo-inc.com

## ABSTRACT

*Contextual advertising* or *Context Match* (CM) refers to the placement of commercial textual advertisements within the content of a generic web page, while *Sponsored Search* (SS) advertising consists in placing ads on result pages from a web search engine, with ads driven by the originating query. In CM there is usually an intermediary commercial *ad-network* entity in charge of optimizing the ad selection with the twin goal of increasing revenue (shared between the publisher and the ad-network) and improving the user experience. With these goals in mind it is preferable to have ads relevant to the page content, rather than generic ads.

The SS market developed quicker than the CM market, and most textual ads are still characterized by "bid phrases" representing those queries where the advertisers would like to have their ad displayed. Hence, the first technologies for CM have relied on previous solutions for SS, by simply extracting one or more phrases from the given page content, and displaying ads corresponding to searches on these phrases, in a purely syntactic approach. However, due to the vagaries of phrase extraction, and the lack of context, this approach leads to many irrelevant ads. To overcome this problem, we propose a system for contextual ad matching based on a combination of semantic and syntactic features.

**Categories and Subject Descriptors:** H.3.3 [Information Storage and Retrieval]: Selection process

**General Terms:** Algorithms, Measurement, Performance, Experimentation

**Keywords:** Contextual Advertising, Semantics, Matching

## 1. INTRODUCTION

Web advertising supports a large swath of today's Internet ecosystem. The total internet advertiser spend in US alone in 2006 is estimated at over 17 billion dollars with a growth rate of almost 20% year over year. A large part of this market consists of *textual ads*, that is, short text messages usually marked as "sponsored links" or similar. The main advertising channels used to distribute textual ads are:

1. *Sponsored Search* or *Paid Search* advertising which consists in placing ads on the result pages from a web search engine, with ads driven by the originating query. All major current web search engines (Google, Yahoo!, and Microsoft) support such ads and act simultaneously as a search engine and an ad agency.

2. *Contextual advertising* or *Context Match* which refers to the placement of commercial ads within the content of a generic web page. In contextual advertising usually there is a commercial intermediary, called an *ad-network*, in charge of optimizing the ad selection with the twin goal of increasing revenue (shared between publisher and ad-network) and improving user experience. Again, all major current web search engines (Google, Yahoo!, and Microsoft) provide such ad-networking services but there are also many smaller players.

The SS market developed quicker than the CM market, and most textual ads are still characterized by "bid phrases" representing those queries where the advertisers would like to have their ad displayed. (See [5] for a "brief history"). However, today, almost all of the for-profit non-transactional web sites (that is, sites that do not sell anything directly) rely at least in part on revenue from context match. CM supports sites that range from individual bloggers and small niche communities to large publishers such as major newspapers. Without this model, the web would be a lot smaller!

The prevalent pricing model for textual ads is that the advertisers pay a certain amount for every click on the advertisement (pay-per-click or PPC). There are also other models used: pay-per-impression, where the advertisers pay for the number of exposures of an ad and pay-per-action where the advertiser pays only if the ad leads to a sale or similar transaction. For simplicity, we only deal with the PPC model in this paper.

Given a page, rather than placing generic ads, it seems preferable to have ads related to the content to provide a better user experience and thus to increase the probability of clicks. This intuition is supported by the analogy to conventional publishing where there are very successful magazines (e.g. *Vogue*) where a majority of the content is topical advertising (fashion in the case of Vogue) and by user studies that have confirmed that increased relevance increases the number of ad-clicks [4, 13].

Previous published approaches estimated the ad relevance based on co-occurrence of the same words or phrases within the ad and within the page (see [7, 8] and Section 3 for more details). However targeting mechanisms based solely

*SIGIR'07,* July 23–27, 2007, Amsterdam, The Netherlands.

on phrases found within the text of the page can lead to problems: For example, a page about a famous golfer named "John Maytag" might trigger an ad for "Maytag dishwashers" since Maytag is a popular brand. Another example could be a page describing the Chevy Tahoe truck (a popular vehicle in US) triggering an ad about "Lake Tahoe vacations". Polysemy is not the only culprit: there is a (maybe apocryphal) story about a lurid news item about a headless body found in a suitcase triggering an ad for Samsonite luggage! In all these examples the mismatch arises from the fact that the ads are not appropriate for the context.

In order to solve this problem we propose a matching mechanism that combines a semantic phase with the traditional keyword matching, that is, a syntactic phase. The semantic phase classifies the page and the ads into a taxonomy of topics and uses the proximity of the ad and page classes as a factor in the ad ranking formula. Hence we favor ads that are topically related to the page and thus avoid the pitfalls of the purely syntactic approach. Furthermore, by using a hierarchical taxonomy we allow for the gradual generalization of the ad search space in the case when there are no ads matching the precise topic of the page. For example if the page is about an event in curling, a rare winter sport, and contains the words "Alpine Meadows", the system would still rank highly ads for skiing in Alpine Meadows as these ads belong to the class "skiing" which is a sibling of the class "curling" and both of these classes share the parent "winter sports".

In some sense, the taxonomy classes are used to select the set of applicable ads and the keywords are used to narrow down the search to concepts that are of too small granularity to be in the taxonomy. The taxonomy contains nodes for topics that do not change fast, for example, brands of digital cameras, say "Canon". The keywords capture the specificity to a level that is more dynamic and granular. In the digital camera example this would correspond to the level of a particular model, say "Canon SD450" whose advertising life might be just a few months. Updating the taxonomy with new nodes or even new vocabulary each time a new model comes to the market is prohibitively expensive when we are dealing with millions of manufacturers.

In addition to increased click through rate (CTR) due to increased relevance, a significant but harder to quantify benefit of the semantic-syntactic matching is that the resulting page has a unified feel and improves the user experience. In the Chevy Tahoe example above, the classifier would establish that the page is about cars/automotive and only those ads will be considered. Even if there are no ads for this particular Chevy model, the chosen ads will still be within the automotive domain.

To implement our approach we need to solve a challenging problem: classify both pages and ads within a large taxonomy (so that the topic granularity would be small enough) with high precision (to reduce the probability of mis-match). We evaluated several classifiers and taxonomies and in this paper we present results using a taxonomy with close to 6000 nodes using a variation of the Rocchio's classifier [9]. This classifier gave the best results in both page and ad classification, and ultimately in ad relevance.

The paper proceeds as follows. In the next section we review the basic principles of the contextual advertising. Section 3 overviews the related work. Section 4 describes the taxonomy and document classifier that were used for page and ad classification. Section 5 describes the semantic-syntactic method. In Section 6 we briefly discuss how to search efficiently the ad space in order to return the top-k ranked ads. Experimental evaluation is presented in Section 7. Finally, Section 8 presents the concluding remarks.

## 2. OVERVIEW OF CONTEXTUAL ADVERTISING

Contextual advertising is an interplay of four players:

- The **publisher** is the owner of the web pages on which the advertising is displayed. The publisher typically aims to maximize advertising revenue while providing a good user experience.
- The **advertiser** provides the supply of ads. Usually the activity of the advertisers are organized around *campaigns* which are defined by a set of ads with a particular temporal and thematic goal (e.g. sale of digital cameras during the holiday season). As in traditional advertising, the goal of the advertisers can be broadly defined as the promotion of products or services.
- The **ad network** is a mediator between the advertiser and the publisher and selects the ads that are put on the pages. The ad-network shares the advertisement revenue with the publisher.
- **Users** visit the web pages of the publisher and interact with the ads.

Contextual advertising usually falls into the category of *direct marketing* (as opposed to *brand advertising*), that is advertising whose aim is a "direct response" where the effect of an campaign is measured by the user reaction. One of the advantages of online advertising in general and contextual advertising in particular is that, compared to the traditional media, it is relatively easy to measure the user response. Usually the desired immediate reaction is for the user to follow the link in the ad and visit the advertiser's web site and, as noted, the prevalent financial model is that the advertiser pays a certain amount for every click on the advertisement (PPC). The revenue is shared between the publisher and the network.

Context match advertising has grown from Sponsored Search advertising, which consists in placing ads on the result pages from a web search engine, with ads driven by the originating query. In most networks, the amount paid by the advertiser for each SS click is determined by an auction process where the advertisers place bids on a search phrase, and their position in the tower of ads displayed in conjunction with the result is determined by their bid. Thus each ad is annotated with one or more *bid phrases*. The bid phrase has no direct bearing on the ad placement in CM. However, it is a concise description of target ad audience as determined by the advertiser and it has been shown to be an important feature for successful CM ad placement [8]. In addition to the bid phrase, an ad is also characterized by a *title* usually displayed in a bold font, and an *abstract* or *creative*, which is the few lines of text, usually less than 120 characters, displayed on the page.

The ad-network model aligns the interests of the publishers, advertisers and the network. In general, clicks bring benefits to both the publisher and the ad network by providing revenue, and to the advertiser by bringing traffic to the target web site. The revenue of the network, given a page $p$, can be estimated as:

$$R = \sum_{i=1..k} P(click|p, a_i) price(a_i, i)$$

where $k$ is the number of ads displayed on page $p$ and $price(a_i, i)$ is the click-price of the current ad $a_i$ at position $i$. The price in this model depends on the set of ads presented on the page. Several models have been proposed to determine the price, most of them based on generalizations of second price auctions. However, for simplicity we ignore the pricing model and concentrate on finding ads that will maximize the first term of the product, that is we search for

$$\arg\max_i P(click|p, a_i)$$

Furthermore we assume that the probability of click for a given ad and page is determined by its relevance score with respect to the page, thus ignoring the positional effect of the ad placement on the page. We assume that this is an orthogonal factor to the relevance component and could be easily incorporated in the model.

## 3. RELATED WORK

Online advertising in general and contextual advertising in particular are emerging areas of research. The published literature is very sparse. A study presented in [13] 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.

A recent work by Ribeiro-Neto et. al. [8] examines a number of strategies to match pages to ads based on extracted keywords. The ads and pages are represented as vectors in a vector space. The first five strategies proposed in that work match the pages and the ads based on the cosine of the angle between the ad vector and the page vector. To find out the important part of the ad, the authors explore using different ad sections (bid phrase, title, body) as a basis for the ad vector. The winning strategy out of the first five requires the bid phrase to appear on the page and then ranks all such ads by the cosine of the union of all the ad sections and the page vectors.

While both pages and ads are mapped to the same space, there is a discrepancy (impendence mismatch) between the vocabulary used in the ads and in the pages. Furthermore, since in the vector model the dimensions are determined by the number of unique words, plain cosine similarity will not take into account synonyms. To solve this problem, Ribeiro-Neto et al expand the page vocabulary with terms from other similar pages weighted based on the overall similarity of the origin page to the matched page, and show improved matching precision.

In a follow-up work [7] the authors propose a method to learn impact of individual features using genetic programming to produce a matching function. The function is represented as a tree composed of arithmetic operators and the *log* function as internal nodes, and different numerical features of the query and ad terms as leafs. The results show that genetic programming finds matching functions that significantly improve the matching compared to the best method (without page side expansion) reported in [8].

Another approach to contextual advertising is to reduce it to the problem of sponsored search advertising by extracting phrases from the page and matching them with the bid phrase of the ads. In [14] a system for phrase extraction is described that used a variety of features to determine the importance of page phrases for advertising purposes. The system is trained with pages that have been hand annotated with important phrases. The learning algorithm takes into account features based on *tf-idf*, html meta data and query logs to detect the most important phrases. During evaluation, each page phrase up to length 5 is considered as potential result and evaluated against a trained classifier. In our work we also experimented with a phrase extractor based on the work reported in [12]. While increasing slightly the precision, it did not change the relative performance of the explored algorithms.

## 4. PAGE AND AD CLASSIFICATION

### 4.1 Taxonomy Choice

The semantic match of the pages and the ads is performed by classifying both into a common taxonomy. The matching process requires that the taxonomy provides sufficient differentiation between the common commercial topics. For example, classifying all medical related pages into one node will not result into a good classification since both "sore foot" and "flu" pages will end up in the same node. The ads suitable for these two concepts are, however, very different. To obtain sufficient resolution, we used a taxonomy of around 6000 nodes primarily built for classifying commercial interest queries, rather than pages or ads. This taxonomy has been commercially built by Yahoo! US. We will explain below how we can use the same taxonomy to classify pages and ads as well.

Each node in our source taxonomy is represented as a collection of exemplary bid phrases or queries that correspond to that node concept. Each node has on average around 100 queries. The queries placed in the taxonomy are high volume queries and queries of high interest to advertisers, as indicated by an unusually high cost-per-click (CPC) price.

The taxonomy has been populated by human editors using keyword suggestions tools similar to the ones used by ad networks to suggest keywords to advertisers. After initial seeding with a few queries, using the provided tools a human editor can add several hundreds queries to a given node. Nevertheless, it has been a significant effort to develop this 6000-nodes taxonomy and it has required several person-years of work. A similar-in-spirit process for building enterprise taxonomies via queries has been presented in [6]. However, the details and tools are completely different. Figure 1 provides some statistics about the taxonomy used in this work.

### 4.2 Classification Method

As explained, the semantic phase of the matching relies on ads and pages being topically close. Thus we need to classify pages into the same taxonomy used to classify ads. In this section we overview the methods we used to build a page and an ad classifier pair. The detailed description and evaluation of this process is outside the scope of this paper.

Given the taxonomy of queries (or bid-phrases – we use these terms interchangeably) described in the previous section, we tried three methods to build corresponding page and ad classifiers. For the first two methods we tried to find exemplary pages and ads for each concept as follows:

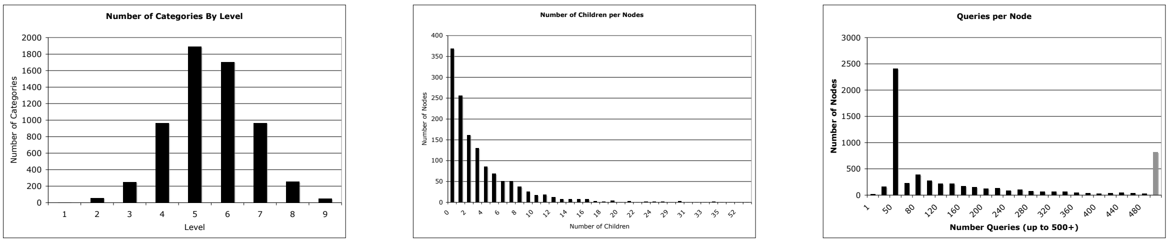

**Figure 1: Taxonomy statistics: categories per level; fanout for non-leaf nodes; and queries per node**

We generated a page training set by running the queries in the taxonomy over a Web search index and using the top 10 results after some filtering as documents labeled with the query's label. On the ad side we generated a training set for each class by selecting the ads that have a bid phrase assigned to this class. Using this training sets we then trained a hierarchical SVM [2] (one against all between every group of siblings) and a log-regression [11] classifier. (The second method differs from the first in the type of secondary filtering used. This filtering eliminates low content pages, pages deemed unsuitable for advertising, pages that lead to excessive class confusion, etc.)

However, we obtained the best performance by using the third document classifier, based on the information in the source taxonomy queries only. For each taxonomy node we concatenated all the exemplary queries into a single meta-document. We then used the meta document as a centroid for a nearest-neighbor classifier based on Rocchio's framework [9] with only positive examples and no relevance feedback. Each centroid is defined as a sum of the *tf-idf* values of each term, normalized by the number of queries in the class

$$\vec{c_j} = \frac{1}{|C_j|} \sum_{\vec{q} \in C_j} \frac{\vec{q}}{\|\vec{q}\|}$$

where $\vec{c_j}$ is the centroid for class $C_j$; $q$ iterates over the queries in a particular class.

The classification is based on the cosine of the angle between the document $d$ and the centroid meta-documents:

$$C_{max} = \arg\max_{C_j \in C} \frac{\vec{c_j}}{\|\vec{c_j}\|} \cdot \frac{\vec{d}}{\|\vec{d}\|}$$

$$= \arg\max_{C_j \in C} \frac{\sum_{i \in |F|} c_j^i \cdot d^i}{\sqrt{\sum_{i \in |F|} (c_j^i)^2} \sqrt{\sum_{i \in |F|} (d^i)^2}}$$

where $F$ is the set of features. The score is normalized by the document and class length to produce comparable score. The terms $c^i$ and $d^i$ represent the weight of the $i$th feature in the class centroid and the document respectively. These weights are based on the standard *tf-idf* formula. As the score of the max class is normalized with regard to document length, the scores for different documents are comparable.

We conducted tests using professional editors to judge the quality of page and ad class assignments. The tests showed that for both ads and pages the Rocchio classifier returned the best results, especially on the page side. This is probably a result of the noise induced by using a search engine to machine generate training pages for the SVM and log-regression classifiers. It is an area of current investigation how to improve the classification using a noisy training set. Based on the test results we decided to use the Rocchio's classifier on both the ad and the page side.

# 5. SEMANTIC-SYNTACTIC MATCHING

Contextual advertising systems process the content of the page, extract features, and then search the ad space to find the best matching ads. Given a page $p$ and a set of ads $A = \{a_1 \ldots a_s\}$ we estimate the relative probability of click $P(click|p, a)$ with a score that captures the quality of the match between the page and the ad. To find the best ads for a page we rank the ads in $A$ and select the top few for display. The problem can be formally defined as matching every page in the set of all pages $P = \{p_1, \ldots p_{pc}\}$ to one or more ads in the set of ads. Each page is represented as a set of page sections $p_i = \{p_{i,1}, p_{i,2} \ldots p_{i,m}\}$. The sections of the page represent different structural parts, such as title, metadata, body, headings, etc. In turn, each section is an unordered bag of terms (keywords). A page is represented by the union of the terms in each section:

$$p_i = \{pw_1^{s1}, pw_2^{s1} \ldots pw_m^{si}\}$$

where $pw$ stands for a page word and the superscript indicates the section of each term. A term can be a unigram or a phrase extracted by a phrase extractor [12].

Similarly, we represent each ad as a set of sections $a = \{a_1, a_2, \ldots a_l\}$, each section in turn being an unordered set of terms:

$$a_i = \{aw_1^{s1}, aw_2^{s1} \ldots aw_l^{sj}\}$$

where $aw$ is an ad word. The ads in our experiments have 3 sections: title, body, and bid phrase. In this work, to produce the match score we use only the ad/page textual information, leaving user information and other data for future work.

Next, each page and ad term is associated with a weight based on the *tf-idf* values. The *tf* value is determined based on the individual ad sections. There are several choices for the value of $idf$, based on different scopes. On the ad side, it has been shown in previous work that the set of ads of one campaign provide good scope for the estimation of idf that leads to improved matching results [8]. However, in this work for simplicity we do not take into account campaigns.

To combine the impact of the term's section and its *tf-idf* score, the ad/page term weight is defined as:

$$tWeight(kw^{si}) = weightSection(S_i) \cdot tf\_idf(kw)$$

where $tWeight$ stands for *term weight* and $weightSection(S_i)$ is the weight assigned to a page or ad section. This weight is a fixed parameter determined by experimentation.

Each ad and page is classified into the topical taxonomy. We define these two mappings:

$$Tax(p_i) = \{pc_{i1}, \dots pc_{iu}\}$$

$$Tax(a_j) = \{ac_{j1} \dots ac_{jv}\}$$

where $pc$ and $ac$ are page and ad classes correspondingly. Each assignment is associated with a weight given by the classifier. The weights are normalized to sum to 1:

$$\sum_{c \in Tax(x_i)} cWeight(c) = 1$$

where $x_i$ is either a page or an ad, and $cWeights(c)$ is the class weight - normalized confidence assigned by the classifier. The number of classes can vary between different pages and ads. This corresponds to the number of topics a page/ad can be associated with and is almost always in the range 1-4.

Now we define the relevance score of an ad $a_i$ and page $p_i$ as a convex combination of the keyword (syntactic) and classification (semantic) score:

$$Score(p_i, a_i) = \alpha \cdot TaxScore(Tax(p_i), Tax(a_i))$$

$$+(1 - \alpha) \cdot KeywordScore(p_i, a_i)$$

The parameter $\alpha$ determines the relative weight of the taxonomy score and the keyword score.

To calculate the keyword score we use the vector space model [1] where both the pages and ads are represented in n-dimensional space - one dimension for each distinct term. The magnitude of each dimension is determined by the $tWeight()$ formula. The keyword score is then defined as the cosine of the angle between the page and the ad vectors:

$$KeywordScore(p_i, a_i)$$

$$= \frac{\sum_{i \in |K|} tWeight(pw_i) \cdot tWeight(aw_i)}{\sqrt{\sum_{i \in |K|} (tWeight(pw_i))^2} \sqrt{\sum_{i \in |K|} (tWeight(aw_i))^2}}$$

where $K$ is the set of all the keywords. The formula assumes independence between the words in the pages and ads. Furthermore, it ignores the order and the proximity of the terms in the scoring. We experimented with the impact of phrases and proximity on the keyword score and did not see a substantial impact of these two factors.

We now turn to the definition of the $TaxScore$. This function indicates the topical match between a given ad and a page. As opposed to the keywords that are treated as independent dimensions, here the classes (topics) are organized into a hierarchy. One of the goals in the design of the $TaxScore$ function is to be able to generalize within the taxonomy, that is accept topically related ads. Generalization can help to place ads in cases when there is no ad that matches both the category and the keywords of the page. The example in Figure 2 illustrates this situation. In this example, in the absence of ski ads, a page about skiing containing the word “Atomic” could be matched to the available snowboarding ad for the same brand.

In general we would like the match to be stronger when both the ad and the page are classified into the same node and weaker when the distance between the nodes in the taxonomy gets larger. There are multiple ways to specify the distance between two taxonomy nodes. Besides the above requirement, this function should lend itself to an efficient search of the ad space. Given a page we have to find the ad in a few milliseconds, as this impacts the presentation to a waiting user. This will be further discussed in the next section.

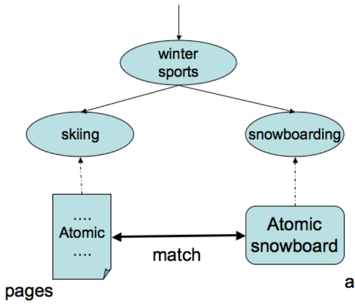

**Figure 2: Two generalization paths**

To capture both the generalization and efficiency requirements we define the $TaxScore$ function as follows:

$$TaxScore(PC, AC) =$$

$$\sum_{pc \in PC} \sum_{ac \in AC} idist(LCA(pc, ac), ac) \cdot cWeight(pc) \cdot cWeight(ac)$$

In this function we consider every combination of page class and ad class. For each combination we multiply the product of the class weights with the inverse distance function between the least common ancestor of the two classes (LCA) and the ad class. The inverse distance function $idist(c1, c2)$ takes two nodes on the same path in the class taxonomy and returns a number in the interval $[0, 1]$ depending on the distance of the two class nodes. It returns 1 if the two nodes are the same, and declines toward 0 when $LCA(pc, ac)$ or $ac$ is the root of the taxonomy. The rate of decline determines the weight of the generalization versus the other terms in the scoring formula.

To determine the rate of decline we consider the impact on the specificity of the match when we substitute a class with one of its ancestors. In general the impact is topic dependent. For example the node “Hobby” in our taxonomy has tens of children, each representing a different hobby, two examples being “Sailing” and “Knitting”. Placing an ad about “Knitting” on a page about “Sailing” does not make lots of sense. However, in the “Winter Sports” example above, in the absence of better alternative, skiing ads could be put on snowboarding pages as they might promote the same venues, equipment vendors etc. Such detailed analysis on a case by case basis is prohibitively expensive due to the size of the taxonomy.

One option is to use the confidences of the ancestor classes as given by the classifier. However we found these numbers not suitable for this purpose as the magnitude of the confidences does not necessarily decrease when going up the tree. Another option is to use explore-exploit methods based on machine-learning from the click feedback as described in [10]. However for simplicity, in this work we chose a simple heuristic to determine the cost of generalization from a child to a parent. In this heuristic we look at the broadening of the scope when moving from a child to a parent. We estimate the broadening by the density of ads classified in the parent nodes vs the child node. The density is obtained by classifying a large set of ads in the taxonomy using the document classifier described above. Based on this, let $n_c$ be the number of document classified into the subtree rooted at $c$. Then we define:

$$idist(c, p) = \frac{n_c}{n_p}$$

where $c$ represents the child node and $p$ is the parent node. This fraction can be viewed as a probability of an ad belonging to the parent topic being suitable for the child topic.

# 6. SEARCHING THE AD SPACE

In the previous section we discussed the choice of scoring function to estimate the match between an ad and a page. The top-k ads with the highest score are offered by the system for placement on the publisher's page. The process of score calculation and ad selection is to be done in real time and therefore must be very efficient. As the ad collections are in the range of hundreds of millions of entries, there is a need for indexed access to the ads.

Inverted indexes provide scalable and low latency solutions for searching documents. However, these have been traditionally used to search based on keywords. To be able to search the ads on a combination of keywords and classes we have mapped the classification match to term match and adapted the scoring function to be suitable for fast evaluation over inverted indexes. In this section we overview the ad indexing and the ranking function of our prototype ad search system for matching ads and pages.

We used a standard inverted index framework where there is one posting list for each distinct term. The ads are parsed into terms and each term is associated with a weight based on the section in which it appears. Weights from distinct occurrences of a term in an ad are added together, so that the posting lists contain one entry per term/ad combination.

The next challenge is how to index the ads so that the class information is preserved in the index? A simple method is to create unique meta-terms for the classes and annotate each ad with one meta-term for each assigned class. However this method does not allow for generalization, since only the ads matching an exact label of the page would be selected. Therefore we annotated each ad with one meta-term for each ancestor of the assigned class. The weights of meta-terms are assigned according to the value of the *idist*() function defined in the previous section. On the query side, given the keywords and the class of a page, we compose a keyword only query by inserting one class term for each ancestor of the classes assigned to the page.

The scoring function is adapted to the two part score - one for the class meta-terms and another for the text term. During the class score calculation, for each class path we use only the lowest class meta-term, ignoring the others. For example, if an ad belongs to the "Skiing" class and is annotated with both "Skiing" and its parent "Winter Sports", the index will contain the special class meta-terms for both "Skiing" and "Winter Sports" (and all their ancestors) with the weights according to the product of the classifier confidence and the *idist* function. When matching with a page classified into "Skiing", the query will contain terms for class "Skiing" and for each of its ancestors. However when scoring an ad classified into "Skiing" we will use the weight for the "Skiing" class meta-term. Ads classified into "Snowboarding" will be scored using the weight of the "Winter Sports" meta-term. To make this check efficiently we keep a sorted list of all the class paths and, at scoring time, we search the paths bottom up for a meta-term appearing in the ad. The first meta-term is used for scoring, the rest are ignored.

At runtime, we evaluate the query using a variant of the WAND algorithm [3]. This is a document-at-a-time algorithm [1] that uses a branch-and-bound approach to derive efficient moves for the cursors associated to the postings lists. It finds the next cursor to be moved based on an upper bound of the score for the documents at which the cursors are currently positioned. The algorithm keeps a heap of current best candidates. Documents with an upper bound smaller than the current minimum score among the candidate documents can be eliminated from further considerations, and thus the cursors can skip over them. To find the upper bound for a document, the algorithm assumes that all cursors that are before it will hit this document (i.e. the document contains all those terms represented by cursors before or at that document). It has been shown that WAND can be used with any function that is monotonic with respect to the number of matching terms in the document.

Our scoring function is monotonic since the score can never decrease when more terms are found in the document. In the special case when we add a cursor representing an ancestor of a class term already factored in the score, the score simply does not change (we add 0). Given these properties, we use an adaptation of the WAND algorithm where we change the calculation of the scoring function and the upper bound score calculation to reflect our scoring function. The rest of the algorithm remains unchanged.

# 7. EXPERIMENTAL EVALUATION

## 7.1 Data and Methodology

We quantify the effect of the semantic-syntactic matching using a set of 105 pages. This set of pages was selected by a random sample of a larger set of around 20 million pages with contextual advertising. Ads for each of these pages have been selected from a larger pool of ads (tens of millions) by previous experiments conducted by Yahoo! US for other purposes. Each page-ad pair has been judged by three or more human judges on a 1 to 3 scale:

1. **Relevant** The ad is semantically directly related to the main subject of the page. For example if the page is about the National Football League and the ad is about tickets for NFL games, it would be scored as 1.

2. **Somewhat relevant** The ad is related to the secondary subject of the page, or is related to the main topic of the page in a general way. In the NFL page example, an ad about NFL branded products would be judged as 2.

3. **Irrelevant** The ad is unrelated to the page. For example a mention of the NFL player John Maytag triggers washing machine ads on a NFL page.

| pages | 105 |
|---|---|
| words per page | 868 |
| judgments | 2946 |
| judg. inter-editor agreement | 84% |
| unique ads | 2680 |
| unique ads per page | 25.5 |
| page classification precision | 70% |
| ad classification precision | 86% |

**Table 1: Dataset statistics**

To obtain a score for a page-ad pair we average all the scores and then round to the closest integer. We then used these judgments to evaluate how well our methods distinguish the positive and the negative ad assignments for each page. The statistics of the page dataset is given in Table 1.

The original experiments that paired the pages and the ads are loosely related to the syntactic keyword based matching and classification based assignment but used different taxonomies and keyword extraction techniques. Therefore we could not use standard pooling as an evaluation method since we did not control the way the pairs were selected and could not precisely establish the set of ads from which the placed ads were selected.

Instead, in our evaluation for each page we consider only those ads for which we have judgment. Each different method was applied to this set and the ads were ranked by the score. The relative effectiveness of the algorithms were judged by comparing how well the methods separated the ads with positive judgment from the ads with negative judgment. We present precision on various levels of recall within this set. As the set of ads per page is relatively small, the evaluation reports precision that is higher than it would be with a larger set of negative ads. However, these numbers still establish the relative performance of the algorithms and we can use it to evaluate performance at different score thresholds.

In addition to the precision-recall over the judged ads, we also present Kendall's $\tau$ rank correlation coefficient to establish how far from the perfect ordering are the orderings produced by our ranking algorithms. For this test we ranked the judged ads by the scores assigned by the judges and then compared this order to the order assigned by our algorithms. Finally we also examined the precision at position 1, 3 and 5.

## 7.2 Results

Figure 3 shows the precision recall curves for the syntactic matching (keywords only used) vs. a syntactic-semantic matching with the optimal value of $\alpha = 0.8$ (judged by the 11-point score [1]). In this figure, we assume that the ad-page pairs judged with 1 or 2 are positive examples and the 3s are negative examples. We also examined counting only the pairs judged with 1 as positive examples and did not find a significant change in the relative performance of the tested methods. Overlaid are also results using phrases in the keyword match. We see that these additional features do not change the relative performance of the algorithm.

The graphs show significant impact of the class information, especially in the mid range of recall values. In the low recall part of the chart the curves meet. This indicates that when the keyword match is really strong (i.e. when the ad is almost contained within the page) the precision of the syntactic keyword match is comparable with that of the semantic-syntactic match. Note however that the two methods might produce different ads and could be used as a complement at level of recall.

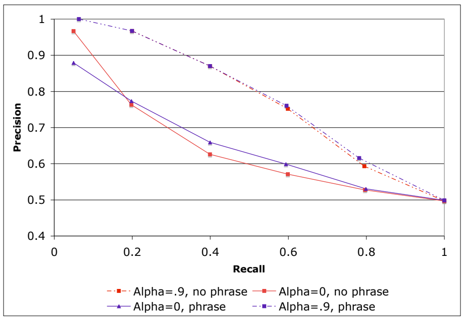

**Figure 3: Data Set 2: Precision vs. Recall of syntactic match ($\alpha = 0$) vs. syntactic-semantic match ($\alpha = 0.8$)**

| $\alpha$ | Kendall's $\tau$ |
|---|---|
| $\alpha = 0$ | 0.086 |
| $\alpha = 0.25$ | 0.155 |
| $\alpha = 0.50$ | 0.166 |
| $\alpha = 0.75$ | 0.158 |
| $\alpha = 1$ | 0.136 |

**Table 2: Kendall's $\tau$ for different $\alpha$ values**

The semantic components provides largest lift in precision at the mid range of recall where 25% improvement is achieved by using the class information for ad placement. This means that when there is somewhat of a match between the ad and the page, the restriction to the right classes provides a better scope for selecting the ads.

Table 2 shows the Kendall's $\tau$ values for different values of $\alpha$. We calculated the values by ranking all the judged ads for each page and averaging the values over all the pages. The ads with tied judgment were given the rank of the middle position. The results show that when we take into account all the ad-page pairs, the optimal value of $\alpha$ is around 0.5. Note that purely syntactic match ($\alpha = 0$) is by far the weakest method.

Figure 4 shows the effect of the parameter $\alpha$ in the scoring. We see that in most cases precision grows or is flat when we increase $\alpha$, except at the low level of recall where due to small number of data points there is a bit of jitter in the results.

Table 3 shows the precision at positions 1, 3 and 5. Again, the purely syntactic method has clearly the lowest score by individual positions and the total number of correctly placed ads. The numbers are close due to the small number of the ads considered, but there are still some noticeable trends. For position 1 the optimal $\alpha$ is in the range of 0.25 to 0.75. For positions 3 and 5 the optimum is at $\alpha = 1$. This also indicates that for those ads that have a very high keyword score, the semantic information is somewhat less important. If almost all the words in an ad appear in the page, this ad is

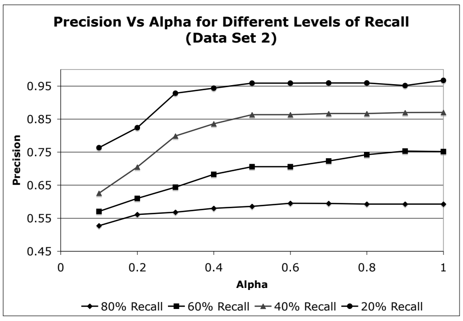

**Figure 4: Impact of $\alpha$ on precision for different levels of recall**

| $\alpha$ | #1 | #3 | #5 | sum |
|---|---|---|---|---|
| $\alpha = 0$ | 80 | 70 | 68 | 218 |
| $\alpha = 0.25$ | 89 | 76 | 73 | 238 |
| $\alpha = 0.5$ | 89 | 74 | 73 | 236 |
| $\alpha = 0.75$ | 89 | 78 | 73 | 240 |
| $\alpha = 1$ | 86 | 79 | 74 | 239 |

**Table 3: Precision at position 1, 3 and 5**

likely to be relevant for this page. However when there is no such clear affinity, the class information becomes a dominant factor.

# 8. CONCLUSION

Contextual advertising is the economic engine behind a large number of non-transactional sites on the Web. Studies have shown that one of the main success factors for contextual ads is their relevance to the surrounding content. All existing commercial contextual match solutions known to us evolved from search advertising solutions whereby a search query is matched to the bid phrase of the ads. A natural extension of search advertising is to extract phrases from the page and match them to the bid phrase of the ads. However, individual phrases and words might have multiple meanings and/or be unrelated to the overall topic of the page leading to miss-matched ads.

In this paper we proposed a novel way of matching advertisements to web pages that rely on a topical (semantic) match as a major component of the relevance score. The semantic match relies on the classification of pages and ads into a 6000 nodes commercial advertising taxonomy to determine their topical distance. As the classification relies on the full content of the page, it is more robust than individual page phrases. The semantic match is complemented with a syntactic match and the final score is a convex combination of the two sub-scores with the relative weight of each determined by a parameter $\alpha$.

We evaluated the semantic-syntactic approach against a syntactic approach over a set of pages with different contextual advertising. As shown in our experimental evaluation, the optimal value of the parameter $\alpha$ depends on the precise objective of optimization (precision at particular position, precision at given recall). However in all cases the optimal value of $\alpha$ is between 0.25 and 0.9 indicating significant effect of the semantic score component. The effectiveness of the syntactic match depends on the quality of the pages used. In lower quality pages we are more likely to make classification errors that will then negatively impact the matching. We demonstrated that it is feasible to build a large scale classifier that has sufficient good precision for this application.

We are currently examining how to employ machine learning algorithms to learn the optimal value of $\alpha$ based on a collection of features of the input pages.

# 9. REFERENCES

[1] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval.* ACM, 1999.

[2] Bernhard E. Boser, Isabelle Guyon, and Vladimir Vapnik. A training algorithm for optimal margin classifiers. In *Computational Learning Theory*, pages 144–152, 1992.

[3] A. Z. Broder, D. Carmel, M. Herscovici, A. Soffer, and J. Zien. Efficient query evaluation using a two-level retrieval process. In *CIKM '03: Proc. of the twelfth intl. conf. on Information and knowledge management*, pages 426–434, New York, NY, 2003. ACM.

[4] P. Chatterjee, D. L. Hoffman, and T. P. Novak. Modeling the clickstream: Implications for web-based advertising efforts. *Marketing Science*, 22(4):520–541, 2003.

[5] D. Fain and J. Pedersen. Sponsored search: A brief history. In *In Proc. of the Second Workshop on Sponsored Search Auctions, 2006.* Web publication, 2006.

[6] S. C. Gates, W. Teiken, and K.-Shin F. Cheng. Taxonomies by the numbers: building high-performance taxonomies. In *CIKM '05: Proc. of the 14th ACM intl. conf. on Information and knowledge management*, pages 568–577, New York, NY, 2005. ACM.

[7] A. Lacerda, M. Cristo, M. Andre; G., W. Fan, N. Ziviani, and B. Ribeiro-Neto. Learning to advertise. In *SIGIR '06: Proc. of the 29th annual intl. ACM SIGIR conf.*, pages 549–556, New York, NY, 2006. ACM.

[8] B. Ribeiro-Neto, M. Cristo, P. B. Golgher, and E. S. de Moura. Impedance coupling in content-targeted advertising. In *SIGIR '05: Proc. of the 28th annual intl. ACM SIGIR conf.*, pages 496–503, New York, NY, 2005. ACM.

[9] J. Rocchio. Relevance feedback in information retrieval. In *The SMART Retrieval System: Experiments in Automatic Document Processing*, pages 313–323. PrenticeHall, 1971.

[10] P. Sandeep, D. Agarwal, D. Chakrabarti, and V. Josifovski. Bandits for taxonomies: A model-based approach. In *In Proc. of the SIAM intl. conf. on Data Mining*, 2007.

[11] T. Santner and D. Duffy. *The Statistical Analysis of Discrete Data.* Springer-Verlag, 1989.

[12] R. Stata, K. Bharat, and F. Maghoul. The term vector database: fast access to indexing terms for web pages. *Computer Networks*, 33(1-6):247–255, 2000.

[13] C. Wang, P. Zhang, R. Choi, and M. D. Eredita. Understanding consumers attitude toward advertising. In *Eighth Americas conf. on Information System*, pages 1143–1148, 2002.

[14] W. Yih, J. Goodman, and V. R. Carvalho. Finding advertising keywords on web pages. In *WWW '06: Proc. of the 15th intl. conf. on World Wide Web*, pages 213–222, New York, NY, 2006. ACM.