Contextual Advertising by Combining Relevance with Click Feedback

Deepayan Chakrabarti
Yahoo! Research
701 First Ave
Sunnyvale, CA 94089.
deepay@yahoo-inc.com

Deepak Agarwal
Yahoo! Research
701 First Ave
Sunnyvale, CA 94089.
dagarwal@yahoo-inc.com

Vanja Josifovski
Yahoo! Research
701 First Ave
Sunnyvale, CA 94089.
vanjaj@yahoo-inc.com

ABSTRACT

Contextual advertising supports much of the Web's ecosystem today. User experience and revenue (shared by the site publisher ad the ad network) depend on the *relevance* of the displayed ads to the page content. As with other document retrieval systems, relevance is provided by scoring the *match* between individual ads (documents) and the content of the page where the ads are shown (query). In this paper we show how this match can be improved significantly by augmenting the ad-page scoring function with extra parameters from a logistic regression model on the words in the pages and ads. A key property of the proposed model is that it can be mapped to standard cosine similarity matching and is suitable for efficient and scalable implementation over inverted indexes. The model parameter values are learnt from logs containing ad impressions and clicks, with shrinkage estimators being used to combat sparsity. To scale our computations to train on an extremely large training corpus consisting of several gigabytes of data, we parallelize our fitting algorithm in a Hadoop [10] framework. Experimental evaluation is provided showing improved click prediction over a holdout set of impression and click events from a large scale real-world ad placement engine. Our best model achieves a 25% lift in precision relative to a traditional information retrieval model which is based on cosine similarity, for recalling 10% of the clicks in our test data.

Categories and Subject Descriptors

H.4.m [**Information Systems**]: Miscellaneous

General Terms

Algorithms, Experimentation, Measurements

Keywords

Clickthrough rate, Modeling, Interaction Effects

1. INTRODUCTION

Web advertising provides financial support for a large portion of today's Internet ecosystem, catering to a diverse set of sites like blogs, news, reviews etc. Spurred by the tremendous growth in traffic in terms of volume, number of users, user engagement, content diversity, the last few years have seen a tremendous growth in spending on web advertising. According to eMarketer [9] the total internet advertiser spending in 2007 will reach almost 20 billion US dollars. This establishes the Web as one of the top 3 advertisement mediums, along with TV and print media.

A major part of the advertising on the web falls into the category of *textual ads*: short textual messages usually marked as "sponsored links" or similar. There are two main types of textual ads on the web today:

1. *Sponsored Search* (SS) or *Paid Search* advertising places ads on the result pages from a web search engine based on the search query. All major current web search engines support such ads and act simultaneously as a search engine and an ad agency.

2. *Contextual advertising* or *Context Match* (CM) advertising places ads within the content of a generic, third-party web page. There usually 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. Here also the main players are the major search engines; however, there are also many smaller players.

While the methods proposed in this paper could be adapted for both SS and CM advertising, we will focus in our analysis and experiments on the CM scenario.

Studies have shown that displaying ads that are closely related to the content of the page[1] provide a better user experience and increase the probability of clicks [4, 21]. This intuition is analogous to that in 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*). Hence, estimating the relevance of an ad to a page is critical in serving ads at run-time.

Previously published approaches estimated the ad relevance based on co-occurrence of the same words or phrases within the ad and within the page (see [18, 13, 3] and the related work section for more details). The model used in this body of work is to translate the ad search into a similarity search in a vector space. Each ad is represented as a vector of features, as for example, unigrams, phrases and classes [3]. The page is also translated to a vector in the

WWW 2008, April 21–25, 2008, Beijing, China.
ACM 978-1-60558-085-2/08/04.

[1] Ads can also be behaviorally targeted; however, without loss of generality, we focus on contextual features.

same space as the ads. The search for the best ads is now translated into finding the ad vectors that are closest to the page vector. To make the search efficient and scalable to hundreds of millions of ads and billions of requests per day, we can use an inverted index and an efficient similarity search algorithm as the one reported in [2]. A drawback of this method is that it relies on a-priori information and does not use the feedback (a posteriori) information that is collected in the form of ad impressions (displays) and clicks.

Another line of work uses click data to produce a CTR estimate for an ad, independent of the page (or query, in the Sponsored Search scenario [19, 17]). The CTR is estimated based on features extracted from the ads that are then used in a learning framework to build models for estimation of the CTR of unseen ads. In this approach the assumption is that the ads are selected by a deterministic method - by matching the bid phrase to a phrase from the page (or the query in Sponsored Search) and therefore to select the most clickable ads we only need to estimate the CTR on the ads with the matching bid phrase. This simplifying assumption of the matching process is an obvious drawback of these approaches. Another drawback is that these methods do not account for differential click probabilities on different pages: If some pages in the corpus attract an audience that clicks on ads significantly more than average, then the learning of feature weights for ads will be biased towards ads that were (only by circumstance) shown on such pages.

Contributions. In this work we combine the two different lines of work and propose a relevance based approach that is augmented to use the the click data to produce a CTR estimate. We adapt the vector space similarity formulation and keep the format of the scoring formulas so that they are still suitable for inverted index evaluation that has low cost and high scalability. To incorporate the feedback, we modify the scoring formulas with correction parameters that are learned from the click data. We examine several scoring formulas in order of increasing complexity and propose learning the parameters by statistical learning methods. We overcome the problem of learning from data with low click rates by a sampling method that equalizes the sizes of clicked and non-clicked portions of the data, and then solves the problem of the (page, ad) sparsity by explaining *click propensities* (scores) of ads on pages in terms of the words that occur in both. In summary our main contributions are:

- We propose a set of intuitive models of click propensities in terms of the words that occur in the webpage and ad. The basic model takes into consideration the location of the words (title, main body, sidebar, etc.), while the most complex model also considers word synonyms, the tf-idf values of the words and relevance measures, among others.

- The models were carefully chosen to extend scoring formulas in current use so that they could be easily implemented on top of existing systems. Thus, they can be immediately used for efficient ad selection from a very large corpus of ads.

- We describe a fast method for fitting the parameters of these models, and prescriptions for picking the right model given the dataset size and runtime execution constraints.

- Extensive experiments on real-world datasets convincingly demonstrate the accuracy of our models.

2. RELATED WORK

Online advertising is an upcoming area of research and the published literature is sparse. A study presented in [21] 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. IR methods used in web search engines are one of the most prominent methods to ensure relevance of search results.

A pioneering report by Ribeiro-Neto et. al on contextual matching [18] examines a number of strategies to match pages to ads based on extracted keywords. Here also the ads and pages are represented as vectors in a vector space. The work first presents five strategies that use cosine similarity to match page and the ad vectors. 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 (impedance 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.

A follow-up work [13] proposes a method to learn impact of individual features using genetic programming to produce a matching function. represented as a tree. Arithmetic operators and the *log* function are internal nodes while different numerical features of the query and ad terms can be leafs of the function tree. 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 [18].

In our previous work [3] we expand on the search of the ads in a vector space by adding classes as additional dimensions. This allows for having the ads topically match the content of the page. We used tf-idf for weighting the unigram and phrase features and class confidence score returned by the classifier as weight of the class features. While the class weights are trained over a labeled set of ads, there is no direct use of the click data to improve on the ad selection.

All of these approaches share the use of similarity in a feature space to relate ads to pages. However none of these approaches uses statistical learning techniques to factor the implicit relevance feedback in a form of click data to learn how to better match pages to ads.

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 [22] 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 use phrases as feature and we learn parameters to determine the relative importance of a particular phrase compared to other phrases and unigrams. Another key difference related to this work is that we use multiple features to select the ads.

Another line of research attempts to predict the click through rate of ads using similar tools (clustering, keyword matching and classification) for search advertising [17]. In this work, the ads are clustered by their bid phrases. The click through rate is averaged over each cluster. The CTR estimate for new ads is obtained by finding the nearest cluster and assuming that cluster's CTR. Results show that this improves over naive global CTR priors and CTR based on the bid phrase deciles.

3. METHOD

Our proposed method to match relevant ads to pages is based on logistic regression, a popular technique in statistics and machine learning [14]. The regression enables us to combine click feedback and semantic information available from both pages and ads to determine relevancy. This is more general than a pure relevance based approach that does not use click feedback in any form. Indeed, our experiments convincingly demonstrate the usefulness of using click feedback to find more relevant ads.

There has been recent work on using regression models for determining relevant ads [19]. While it has the same flavor as our work, only ad-specific features are learnt, which is only a subset of the features we consider. In particular, in addition to page and ad specific features, we learn features that capture *interactions* between pages and ads. Furthermore, we combine word based features with traditional relevance measures to enhance matching relevant ads to pages.

Our models are more granular and can incorporate larger number of features, which reduces bias in CTR estimates and leads to better performance. However, reduced bias comes at the price of increased variance, which can become a serious problem if the models become too granular and start overfitting the training data. To balance these two issues, we use a two-pronged strategy. First, we use a relatively large but *specially selected* set of features, where the selection mechanism ensures that the features have reasonable support. We also provide a mechanism based on prior probabilities to down-weight features that are too sparse. The second strategy we use to prevent overfitting is to train our models on an extremely large corpus (billions of records, several thousand features) which automatically increases the support of a large number of features. Fortunately, data is plentiful especially for big ad-networks that serve a large number of publishers and advertisers. However, increased training size poses a difficult computational challenge of scaling logistic regression to web scale data. We overcome this by using an approximation based on a "divide and conquer strategy", i.e., we randomly split our training corpus into several pieces and fit a separate logistic regression to each piece. The final result is obtained by combining estimates from all the pieces. Our computation is carried out in the software framework called MapReduce[12] that supports large scale parallel computations using a cluster of commodity personal computers.

Roughly speaking, our method consists of three broad steps: (a) Feature extraction, (b) Feature selection, and (c) Coefficient estimation for features through a logistic regression. We provide a detailed description of each below.

3.1 Feature Extraction

Pages and ads are treated as being composed of several *regions*. For instance, a page is composed of page title, page metadata, page body, page URL etc. Similarly, an ad is composed of ad title, ad body etc. Within each region, we extract a set of words/phrases after stop word removal. We associate a score (e.g. region specific tf, tf-idf) to each word that measures its importance in a given region. For a given (page, ad) region combination, our model has three sets of features described below.

Page region specific *main effects.* Web pages are usually composed of multiple regions with different visibility and prominence. The impact of each region on the ad selection can thus vary. We follow the intuition of our previous work [3] and we learn the effect of each region separately. For a word w in page region $p(r)$ with score $t_{p(r)w}$, the region-specific main effect is defined as

$$M_{p(r)w} = 1(w \in p(r)) \cdot t_{p(r)w},$$

that is, if the word is present in the page region $p(r)$, the feature contributes its score else it does not contribute. These features provide an estimate of word popularity. They are not useful at the time of selecting relevant ads for a given page but help in getting better estimates of other terms in the model after adjusting for the effect of popular words on a page. For instance, if "camera" pages are popular in terms of click-through rates and 90% of our corpus consists of camera pages, "camera" ads that were the ones mostly shown on camera pages would tend to become popular even on "soccer" pages which constitute only 1% of the total corpus. By incorporating page words in the model, we adjust for this effect and get the correct matching ads for "soccer" pages.

Ad region specific *main effects.* Ads are also composed of multiple regions, some visible to the user (title, abstract) and some used only in the ad selection (bid phrase, targeting attributes). As with the page regions, the ad regions can have different impact on the ad selection. For a word w in ad region $a(r)$ with score $t_{a(r)w}$, this is defined as

$$M_{a(r)w} = 1(w \in a(r)) \cdot t_{a(r)w}.$$

Unlike page specific main effects, it does play an important role when selecting relevant ads for a given page and provides more weight to popular ads.

***Interaction effects* between page and ad regions.** For a word w_1 in page region $p(r_1)$ and word w_2 in ad region $a(r_2)$ with score $f(t_{p(r_1)w_1}, t_{a(r_2)w_2})$ for some function f, this is given as

$$\begin{aligned} I_{p(r_1)w_1,a(r_2)w_2} &= 1(w_1 \in p(r_1), w_2 \in a(r_2)) \qquad (1) \\ &\cdot \; f(t_{p(r_1)w_1}, t_{a(r_2)w_2}). \end{aligned}$$

In this paper, we confine ourselves to the case where $w_1 = w_2$ (i.e., the feature "fires" only if the same word occurs in both the corresponding page and ad regions), but in general, one can consider co-occurrences of synonyms or related words. Examples of f include the product function

$t_{p(r_1)w_1} \times t_{a(r_2)w_2}$, the geometric mean $\sqrt{t_{p(r_1)w_1} \times t_{a(r_2)w_2}}$ and so on. Interaction effects are important components of our method and help in matching relevant ads to a given page. For instance, occurrence of the word "camera" in the ad body is a strong indication of the ad being relevant for the page whose title contains the word "camera," with the degree of relevance being determined by the regression.

3.2 Feature Selection

For any given (page, ad) region combination, a large number of words occur in the training data. Using them all as features might make the logistic regression ill-conditioned and inflate variance of the coefficient estimates. Hence we take recourse to variable selection techniques which select a subset of important words to be used in our regression. Variable selection in the context of regression is a well studied area with a rich literature. Stepwise backward-forward automated variable selection algorithms are widely used for large scale applications but these methods have drawbacks, especially when features are correlated [8]. The general recommendation is to use as much domain knowledge as possible instead of using an automated procedure to select relevant variables. However, in large scale settings as ours, some level of automation is necessary. For reasons of scalability, we explore a two-stage approach. In the first stage, we conservatively prune non-informative features using simple measures that can be computed using only a few passes over the training corpus. In the second stage, we fit a regression to all the selected features from the first stage but down-weight them through a specially constructed prior that pools data from all the features, while putting more coefficient on those that are less sparse. The latter approach will be described in more detail in the next subsection; we discuss our variable selection methods next.

We selected the variables using two methods, the first based on clicks and views, and the second based on relevance scores of words that are independent of any click feedback. In the first approach (data-based), we rank words based on a measure that quantifies the interaction between words occurring in the page and ad regions. For a word w, the interaction measure is defined as

$$i_w = \frac{CTR_w^{both}}{CTR_w^{page} \cdot CTR_w^{ad}} \qquad (2)$$

where CTR_w^{both} denotes the click-through rate (CTR) when w occurred both on page region and ad region of an ad displayed on a page, and CTR_w^{page} and CTR_w^{ad} denote the marginal CTRs when w shown on the page and ad regions respectively. Higher values of the ratio indicates stronger interaction being induced by the presence of the word which in turn should enhance the matching quality of ads to pages. We also tried a variation of the measure above with a square root of the denominator with no significant impact.

In the second approach (relevance-based), words are ranked by computing the average tf-idf scores across the entire page and ad corpus for the respective regions under consideration. Here, we experiment with two measures: (a) create a single corpus by treating page and ad regions as documents and compute a single tf-idf average score for each word, and (b) Treat the page and ad regions as different corpora and use the geometric mean of tf-idf scores computed separately from page and ad regions for each word.

For both measures, we picked the top 1000 words and used them in the logistic regression. To avoid noisy estimates of CTRs in the ratio, we only consider words that were shown simultaneously on ad and page regions at least 10 times and had non-zero marginal probabilities. As our experiments will show later, the data-based approach gives better results for the same number of words.

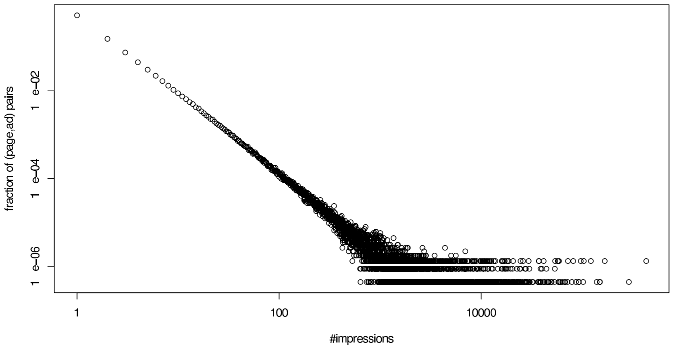


Figure 1: *Distribution of (page, ad) impressions:* Plots are on log-log scale but ticks are on the original scale.

3.3 Approximate Logistic Regression

Let y_{ij} denote the binary click outcome (1 for click, 0 for no click) when ad j is shown on page i. We assume y_{ij} has a Bernoulli distribution with CTR p_{ij}, i.e., the probability distribution of y_{ij} is given by $P(y_{ij}) = p_{ij}^{y_{ij}}(1-p_{ij})^{1-y_{ij}}$. To determine relevant ads for a given page i, we need to estimate p_{ij}'s, with higher values indicating more relevant ads. For ads that are shown a large number of times on a page, the CTR can be estimated empirically by clicks per impression. However, in our application a large fraction of page-ad pairs having a small number of impressions (Figure 1). In fact, since the CTRs are typically low ($0.1\% - 20\%$ with a substantial right skewness in the distribution), number of impressions required to get precise empirical estimates are high. For instance, to estimate a 5% CTR, we need $1K$ impressions to be even 85% confident that our estimate is within 1% of the true CTR. Thus, we take recourse to feature based models, i.e., p_{ij} is a function of features extracted from page and ad regions as discussed in section 3.1.

To allow for arbitrary real-valued coefficients for features, it is routine to map p_{ij} onto the real line via a monotonically increasing function. The most widely used function is the *logit* which maps p_{ij} to $logit(p_{ij}) = \log[p_{ij}/(1-p_{ij})]$. We assume that $logit(p_{ij})$ is a linear function of features representing the main effects and interaction effects discussed in section 3.1. For simplicity, consider a single (page, ad) region combination $(p(r_1), a(r_2))$. The linear function in the logistic regression is given by:

$$\begin{aligned} logit(p_{ij}) &= logit(q_{ij}) + \sum_w \alpha_w M_{p(r_1)w} \qquad (3) \\ &+ \sum_w \beta_w M_{a(r_2)w} + \sum_w \delta_{w,r_1,r_2} I_{p(r_1)w,a(r_2)w} \end{aligned}$$

where $\mathbf{w} = (\alpha, \beta, \delta)$ are unknown feature coefficients to be estimated by logistic regression, and $logit(q_{ij})$ are known *prior* log-odds that could have been derived from a different model. For instance, a uniform prior would assume $q_{ij} = \hat{p}$, where $\hat{p}$ is the average CTR on the entire training corpus. Another possibility we explore is to derive prior log-odds q_{ij} by combining relevance scores with click feedback.

To add new (page,ad) region combination, we only need to augment equation 3 with the appropriate linear terms for the page main and ad main effects. For the interaction effects, we re-parametrize our model to facilitate indexing. We explain our re-parametrization here and discuss the connection to indexing later in section 4. For each (page,ad) combination (r_1, r_2), a word w that occurs in both r_1 and r_2 has a coefficient δ_{w,r_1,r_2} which depends on the word, the page region and the ad region. We assume the following parametrization.

$$\delta_{w,r_1,r_2} = \delta_w \cdot \gamma_{p(r_1)} \cdot \gamma_{a(r_2)} \tag{4}$$

i.e., the interaction of a word for a given page and ad region combination is factored into word-specific, page-specific and ad-specific components. Thus, for M words, R_1 page regions, R_2 ad regions, the number of parameters equals $M + R_1 + R_2$ as opposed to $M \cdot R_1 \cdot R_2$ in the original model. The estimate of coefficients is obtained by maximizing the log-likelihood of the data as given by

$$\sum_{ij} (y_{ij} log(p_{ij}) + (1 - y_{ij}) log(1 - p_{ij})) \tag{5}$$

where p_{ij} is given by equation 3. The optimization problem described above may become ill-conditioned and lead to high variance estimates if features tend to be correlated or are sparse or both. For exact technical details on necessary and sufficient conditions that ensure convergence, we refer the reader to [15]. This is a drawback in our scenario where feature sparsity and correlations are routine. To provide a robust solution, we put additional constraints on the coefficients in the form of priors.

A $N(0, \sigma^2)$ prior would mean that the parameter estimates are *pinned* down in the range $(-3\sigma, 3\sigma)$ with 99% probability a-priori. In the absence of enough information about the coefficient from data, this ensures that the coefficient estimates do not diverge to the boundaries and cause numerical instability. To put more stringent constraints on sparse features, we down-weight the prior variance σ^2 by a measure of relative sparsity which we define to be the variance of the feature occurrence process relative to average feature occurrence variance. The feature occurrence variance is given by $s(1 - s)$, where s is the fraction of times the feature occurs. In particular, we assume:

$$\begin{aligned}
\alpha_w &\sim N\left(0, \sigma^2 \cdot \frac{s_p(w)(1 - s_p(w))}{s_p(1 - s_p)}\right) \\
\beta_w &\sim N\left(0, \sigma^2 \cdot \frac{s_a(w)(1 - s_a(w))}{s_a(1 - s_a)}\right) \\
\delta_w &\sim N\left(0, \sigma^2 \cdot \frac{s_I(w)(1 - s_I(w))}{s_I(1 - s_I)}\right)
\end{aligned}$$

Note that separate averages are used for the main page and ad effects, and interaction effects (indicated by the subscripts p, a, and I). In all our experiments, we fix $\sigma^2 = 9$; experiments with several other values in the range of 3 to 20 did not yield much difference.

Now, the optimization problem reduces to estimating the coefficients by maximizing the log-posterior which is the sum of the log-likelihood (Eq. 5) and the log-prior of the coefficients, as discussed above. Next, we discuss the optimization process itself.

Several approaches to optimize our objective function exist in the literature. Among the ones that have been used in large-scale applications are iterative scaling [20], nonlinear conjugate gradient, quasi-Newton (in particular, limited memory BFGS) [7], iteratively-reweighted least squares [14], truncated Newton [16], and trust-region Newton [5]. All the methods are iterative and generate a sequence of estimates that converge to the optimal solution. For all methods except iterative scaling, cost per iteration is high but the convergence is fast. For iterative scaling which updates one component at a time, cost per iteration is low but convergence is slower. For our application, the training corpus typically has several millions data points and several thousand features making it extremely slow to fit the model using these approaches on a single machine.

To scale our computations, we adopt a simple parallelization approach that randomly splits the data into several parts, fits a logistic regression separately to each part and then combines the estimates obtained from each piece. For convenience, we perform our computation in a MapReduce framework [12]. MapReduce is a programming model for processing large data sets. It runs on a large cluster of commodity machines; it is highly scalable processing several gigabytes of data on thousands of machines and easy to use. The run-time system automatically takes care of the details of partitioning the data, scheduling job across machines, handling failures and managing inter-machine communication.

To fit a logistic regression for a given piece, we use a simple iterative scaling (also known as conditional maximization) approach. The algorithm is as follows: We initialize the coefficients α's, β's, and δ's to 0, and $\gamma_{p(.)}$'s and $\gamma_{a(.)}$'s to 1. We update the value of each coefficient one at a time holding the others fixed at the current value by maximizing the likelihood through a Newton-Raphson method. This completes a single iteration. The procedure is continued through several iterations until convergence. The method is guaranteed to converge since every step can only increase the likelihood. Along with a coefficient estimate, the Newton-Raphson procedure provides an estimate of the negative Hessian, the inverse of which provides an estimate of variance of the coefficient from maximum likelihood theory. The results on the various data partitions are combined using a weighted average of the individual estimates, where the weight assigned to partition-specific estimate is its relative precision obtained from the negative Hessian values. This weighting scheme is the best way to combine estimates through a linear function [6]. Figure 2 describes our fitting procedure in detail.

4. AD SEARCH PROTOTYPE

A key feature of the model proposed in this paper is that it is suitable for efficient evaluation over an inverted indexes. In this section we outline an implementation of a prototype ad search engine based on the WAND [2] algorithm and inverted indexing of the ads using the Hadoop distributed computing framework [10].

Our method allows for any kind of feature to be used in the ad search. In our prototype we use unigrams, phrases and classes as features. The inverted index is composed of one

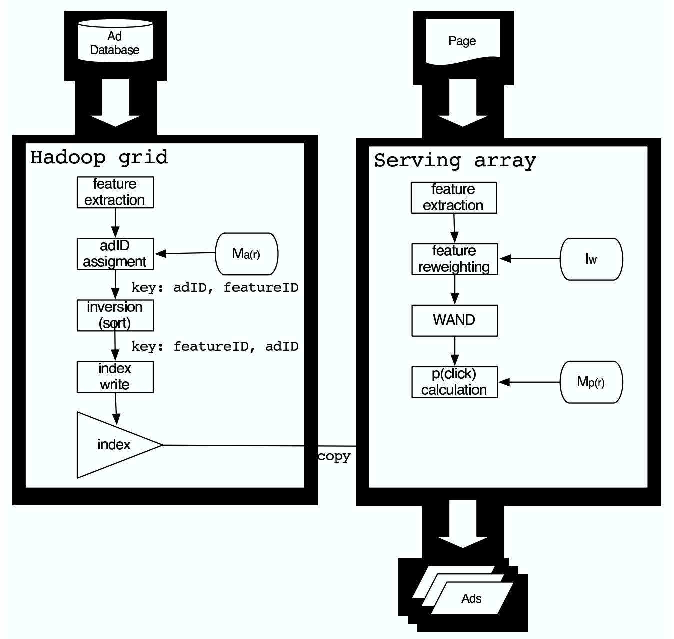


Figure 3: Ad search prototype architecture

postings list for each feature that has one entry (posting) for each ad that contains this feature. The ads are represented by adIDs - unique numeric identifiers assigned to each ad.

Figure 3 shows the architecture of our ad search prototype. We produce the inverted index over a grid of machines running the Hadoop framework [10]. The indexing starts by extracting features from the ads. Each feature is represented by an unique numeric featureID. The resulting data file is sorted by $< adID, featureID >$. Next we invert this file by sorting the file by $< featureID, adID >$ as a key. Finally we write the delta compressed posting lists used at runtime to evaluate the query.

There are a few important differences in the ad search engine problem that require different approach compared to web search engines. First, in web search, the queries are short and the documents are long. In the ad search case, the number of features per ad is usually lower than the number of features extracted from a web page, which in our case represent the ad space query. So it is almost never the case that an ad will contain all the features of the ad search query. Therefore the ad search engine performs similarity search in the vector space with a long query and relatively short ad vectors. In contrast for the majority of the web queries there are many pages that contain all the query words and one of the key issue is how to rank the pages containing the query.

Features are extracted from the input query. The query is a bag of pairs $< featureID, weight >$. For each query feature, WAND opens a *cursor* over the posting list of this feature. During the evaluation the cursors are moved forward examining the documents as they are encountered. WAND is a document at the time algorithm [1] that 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 candidates. The invariant of the algorithm is that the heap contains the best

```
Algorithm CondMax
Initialize $\text{coeff}_i = 0$ for all word features $i$ and $\text{coeff}_i = 1$
for page-region and ad-region specific parameters in the
re-parametrized model of equation 4.
Iterate until convergence
    Iterate $i$ over the set of features
        Set $f'(i) = f''(i) = \text{loglik} = 0$
        Iterate over all impressions, with the data
        providing $\text{score}_i$ for feature $i$
            $x = \sum_i \text{coeff}_i \cdot \text{score}_i$
            $p = 1.0/(1.0 + exp(-x))$
            $f'(i) = f'(i) + (\text{click\_or\_not} - p) \times \text{score}_i$
            $f''(i) = f''(i) - p(1-p) \times \text{score}_i^2$
            $\text{loglik} = \text{loglik} + \log p - x \cdot (1 - \text{click\_or\_not})$
        $f'(i) = f'(i) - \text{coeff}_i/\sigma_i^2$
        $f''(i) = f''(i) - 1.0/\sigma_i^2$
        $\text{coeff}_i = \text{coeff}_i - f'(i)/f''(i)$ if loglik is lower
        than previous
    Return $(\text{coeff}_i, f''(i))$ for all $i$
```

```
Algorithm CombineCoeffs
Randomly split the data
Call CondMax on each split
Iterate over all features $i$
    $C(i) = \frac{\Sigma_{\text{split}_j} \text{coeff}_i \cdot |f''(i)|}{\Sigma_{\text{split}_j} |f''(i)|}$
Return $C(i)$ as the final coefficient value for feature $i$
```

Figure 2: *Implementation of Logistic Regression.*

matches (highest scores) among the documents (ads) with IDs less than the document pointed by the current minimum cursor.

Cursors pointing on documents with upper bound smaller than the minimum score among the candidate docs are candidates for a move. To find the upper bound for a document, the algorithm assumes that all cursors that are before the current 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. It can also be easily shown that some non-monotonic scoring functions can also be used as long as we can find a mechanism to estimate the score upper bounds.

One family of such functions is a set of functions where a fixed subset of the features (known a priori) always decrease the score. In such cases, the upper bound estimates just assume that these features do not appear in the ad. An example of such function is a cosine similarity where some of the query coefficients are negative. The scoring function proposed in this paper might have such coefficients and fits well within the WAND framework.

Incorporating the logistic-regression based model in this framework is simple. The scoring equation 3 is modified to exclude the page effect and used as WAND scoring formula. Figure 3 shows the stages in which the learned parameters are factored into the evaluation framework. $M_{a(r)}$ is used at indexing time to calculate a static score for each individual ad. We use this score to assign an adID to the ads in decreasing ad score order. This allows for estimating upper bounds of the ads that are skipped by using scores by using the score of the ad pointed by the preceding cursor in the sorted cursor list.

After the page is parsed and the features are extracted along with their TF-IDF scores, we apply the reweighing based on the table I_w.

The $M_{p(r)}$ table is not used in the ad selection but just to adjust the final scores to calculate the probabilities according to equation 3.

5. EXPERIMENTS

We now provide a detailed discussion of all our experiments, conducted on a large-scale real-world dataset. After a brief description of the data, we present results on the different word selection schemes, followed by comparisons of several variants of our model with a baseline model that does not utilize any click feedback and is based solely on relevance. The results convincingly demonstrate the improved performance of models that use click feedback.

5.1 Data

To demonstrate the efficacy of our approach, we conduct a series of experiments on retrospective data collected from some back-end servers of a large contextual advertising system. We would like to emphasize that the servers were chosen subjectively and are not representative of the actual performance of the entire advertising system. We conducted our experiments on 30 days of data. The first 15 days of data were used as our training corpus while the results were evaluated on the remaining days.

Every display of an ad on a page is called an impression. Thus, a single page view by a user generates multiple impressions and equals the number of ads shown on the page. There are approximately $.5B - 1B$ page views on a day in the portion of the data that we examined. Defining CTR as the number of clicks per impressions, overall CTRs for (page,ad) combinations are low and approximately in the range of $.1\% - 20\%$ with extreme right skew. Labeling the clicks as 1 and non-clicks as 0, we have a learning problem with extreme class imbalance. The logistic regression is known to provide biased estimates of weights in such scenarios [11]. A widely used strategy in such settings reduces imbalance by sampling from the majority class. We also adopt a similar strategy but our sampling scheme is as follows: we retain all impressions associated with clicked page views and get a 1/2000 random sample from the pageviews associated with non-clicked page views. All results reported in this paper are on data sampled in this fashion.

5.2 Word selection schemes

We conducted experiments with several (page,ad) region combinations but provide detailed analysis for experiments conducted with one region combination, viz, page title (call it *pti*) and ad title (call it *oti*). In the training corpus, we obtained approximately $112K$ unique words after stop word removal. To avoid introducing extremely sparse words in our regression, we truncated our word list to consider only those that had at least 100 impressions where the word occurred both on the page and ad. Also, we further truncated our list by considering only words that had at least 5 clicks separately when shown on pages and ads. The truncation left us with approximately $3.4K$ words to choose from.

We implemented one data based and two relevance-based feature selection schemes. The data based feature selection criteria was described in section 3.1 and given by the measure defined as i_w in equation 2. We conduct two experiments, one with the top $1K$ words and other with all $3.4K$ words. Similar experiments were also conducted on words selected using two relevance-based measures described previously in Section 3.2. Briefly, the first relevance-based measure assigns the average tf-idf score to a word. It is computed from a single corpus which is the union of page title and ad title regions. The second relevance-based measure is the geometric mean of average tf-idf scores of a word computed separately from page region and ad region corpora. We also conducted a set of experiments by randomly selecting a set of $1K$ words from our list of words which performed very poorly and hence its results are not reported.

5.3 Hybrid model

A pure relevance-based based model finds relevance by using semantic information. More specifically, the relevance of an ad for a page is quantified by a score which is a function of page and ad content. We tested two such scores for region combination (*pti*, *oti*), viz.

$$Relevance_1 = \sum_w \text{tf-idf}_{w,pti} \cdot \text{tf-idf}_{w,oti}$$

and

$$Relevance_2 = \frac{Relevance_1}{\sqrt{\sum_w \text{tf-idf}^2_{w,pti} \cdot \sum_w \text{tf-idf}^2_{w,oti}}}$$

$Relevance_2$ is the cosine of the angle between tf-idf vectors for page and ad regions. The distribution of $Relevance_1$

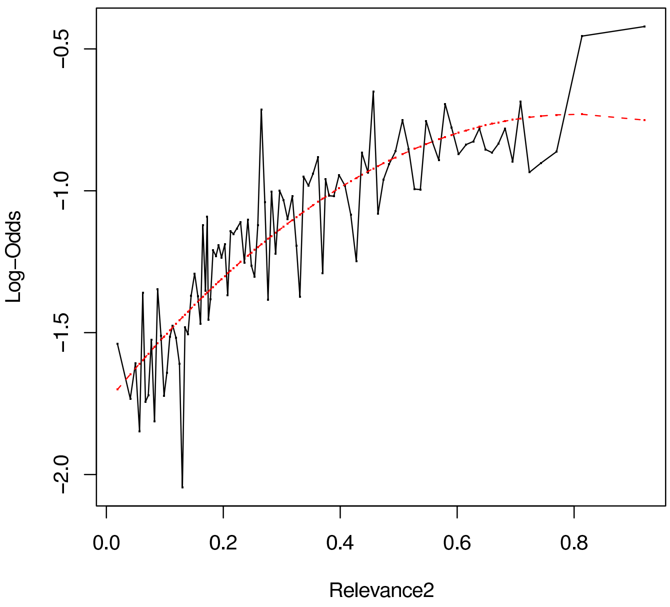


Figure 4: *Relationship between log-odds of CTR and $Relevance_2$.*

was extremely right skewed with several outliers, hence we conduct all our experiments with $Relevance_2$.

In addition to a model based solely on $Relevance_2$, we explore a hybrid model which combines our word-based logistic regression with the relevance-based score model. We explore two options: (a) use $Relevance_2$ as an additional feature in the logistic regression but without adding substantial overhead to the indexing algorithm, and (b) use $Relevance_2$ to derive the priors q_{ij} in equation 3.

Both these options require understanding the relationship between the log-odds of CTR and $Relevance_2$; the relevance score needs to be transformed on to the scale of log-odds of probability. We empirically studied this relationship by plotting empirically estimated log-odds against $Relevance_2$ scores as follows: sort all positive $Relevance_2$ values in the training data and create 100 bins. For each bin, we compute the empirical log-odds and the average $Relevance_2$ score. Figure 4 shows the relationship. Approximately 90% of impressions had a $Relevance_2$ value of zero. Excluding those, the curve clearly reveals a quadratic relationship. Thus, for scheme (a) above, we add quadratic terms $a + b * Relevance_2 + c * Relevance_2^2$ to the logistic regression and estimate parameters a, b and c. For scheme (b) above, we derive q_{ij}'s by using the approximate quadratic curve shown in figure 4:

$$logit(q_{ij}) = -1.75 + 2.52 * Relevance_2 - 1.56 * Relevance_2^2.$$

5.4 Evaluation Criteria

Our test data had a total of 14.48M impressions. Of these, $12.8M$ had at least one word from our $1K$ word list or a positive tf-idf score when considering only words in *pti* and *oti*. We only report results on this pool. We use the precision-recall metric to measure the performance of our algorithms. In our scenario, recall for a given threshold on the model score is the fraction of all clicks that were present (recalled) in the test cases above that threshold. Precision for a given recall value is the CTR on the test cases above the threshold. Since our end goal is to rank ads for a given page, we are more interested in algorithms that recall the top fraction of clicks (e.g 2%, 5%, 10%) with greater precision. Also note that, for a random classifier that classifies a test case as a click with a constant probability (overall CTR in training corpus), the precision equals that constant probability value for every recall value.

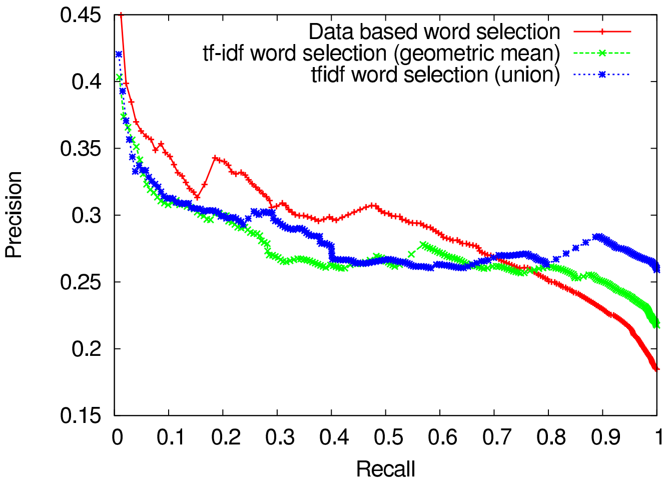


Figure 5: *Precision-recall curves for the three word selection schemes with the best hybrid model that uses both words and relevance measure as features.*

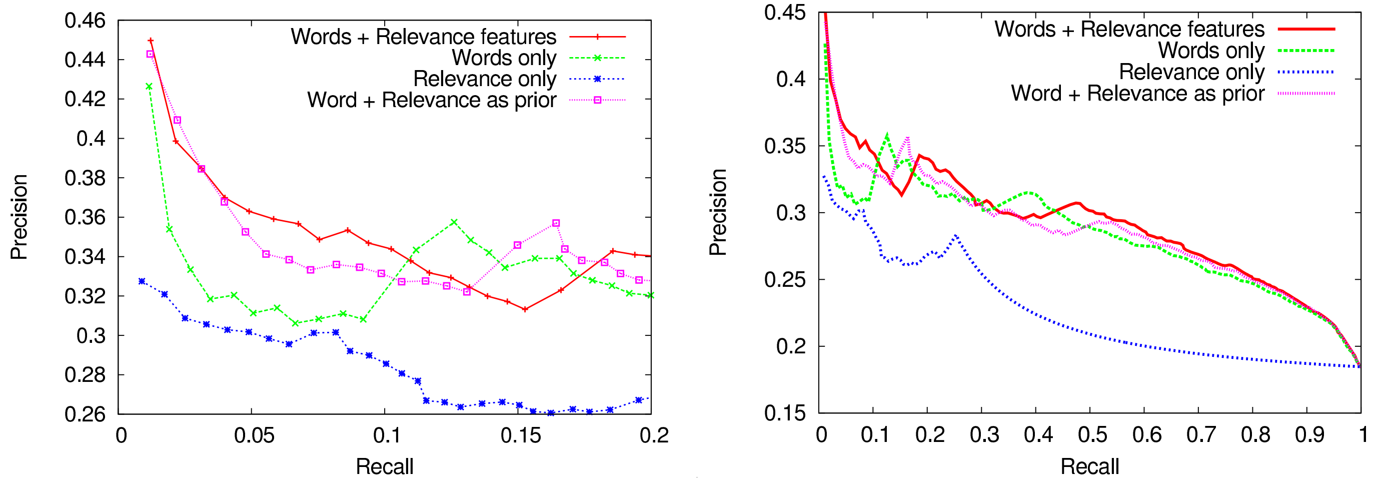
5.5 Results

Our key results are as follows:

- Figure 5 shows the precision-recall curves for the three word selection schemes for our best model, viz, the hybrid model that uses both words and relevance measure as features (results with other models were similar). Of the three word selection schemes we tried, the data-based one using measure i_w of equation 2 gave the best results, especially for low values of recall. Of the two relevance-based word selection criteria, the one that uses the geometric mean of the average tf-idf score was slightly better than the one that used average tf-idf from the union of page and ad regions. However, both the relevance schemes were inferior to the data-based one. In addition, we found no significant difference in going from the top 1000 words to the top 3400 words in the data-based scheme; hence, those results are not shown. This does indicate, however, that the top few words according to our data-based measure i_w are enough to capture most of the signal in the data.

- We conducted experiments using both tf and tf-idf as our scores for features in equation 3. We consistently get better results with tf; all our subsequent results are reported with this measure.

- Figure 6 shows the precision-recall curve for the hybrid, word-only and relevance-based models. Both our word-only logistic regression using tf as feature scores



(a) Truncated at 20% recall

(b) All recall values

Figure 6: ***Precision recall curves for three methods. All curves are reported for the best word selection scheme. The second curve was truncated at a recall value of* 20% *to provide a better summary of performance at small recall values. Statistical error in all cases was less than* 1% *of the mean.***

in equation 3 and the hybrid model that uses both word-based and $Relevance_2$ based features outperformed the pure relevance model that ranks ads solely based on $Relevance_2$. In fact, the hybrid model is superior to the word only model; especially for low recall values like 2%, 5% and 10%. The hybrid models that use $Relevance_2$ as features and prior have approximately similar performance. In fact, for a recall of 10%, the hybrid model provides 25% lift relative to a pure relevance-based model and 100% lift relative to random model in terms of precision. We notice sharp spikes in the precision-recall curves, especially at low recall values. This is indicative of high click density regions in the test data when ranked by model scores. On further scrutiny we discovered some of these high density regions were caused by (page,ad) impressions that had extremely high click rates (e.g in the range of $30\% - 60\%$). This can occur in our data due to a number of reasons: popular ads, popular pages, sudden burst of heavy clickers, etc. The 95% confidence intervals obtained by producing precision-recall curves on several independent splits of test data was within 1% of the mean; hence, all our results are statistically significant.

- Figure 7 shows histograms of parameter estimates obtained from our best hybrid model. The histograms show the distribution of coefficient estimate × average-tf for words for page main effects, ad main effects and page-ad interaction effects. Interestingly, the interaction effects are small for a majority of words except for a small fraction that have large positive values. This ties in with the observation that using the top 1000 words was as accurate as using the top 3400 words; if only a few words have significant interaction effects, then adding more words as features into the logistic regression will not help.

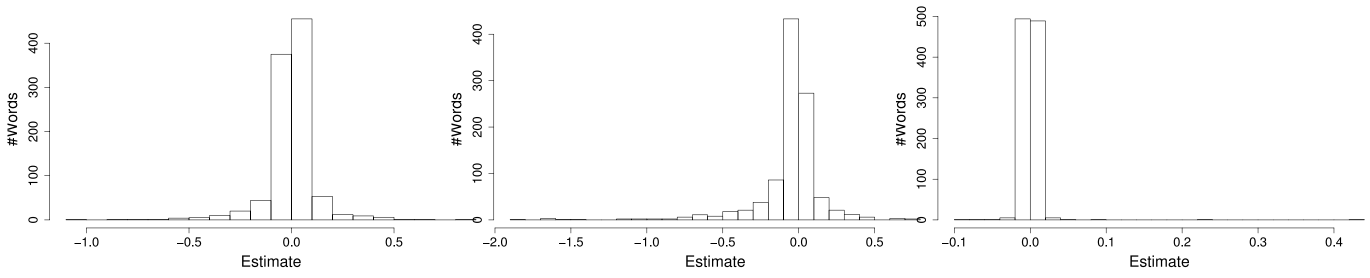
5.6 Summary of Results

Based on an extensive set of large scale experiments, we demonstrated that using click feedback significantly improves the accuracy of matching relevant ads to pages compared to traditional relevance scoring models that are solely based on semantic similarity. We incorporate click feedback through a logistic regression by using the words on pages and ads as features. We tested several variable selection schemes and found a simple data based scheme performs the best. We also proposed a hybrid model which uses words and relevance scores as features and significantly outperforms the traditional relevance based models in our experiments. Our solution is scalable and can process several gigabytes of data through a Hadoop parallelization framework. In addition, it can be easily incorporated into existing ad-serving architectures.

6. CONCLUSIONS

We proposed a new class of models to combine relevance with click feedback for a contextual advertising system. Our model is based on a logistic regression and allows for a large number of granular features. The key feature of our modeling approach is the ability to model interactions that exist among words between page and ad regions in a way that is suitable for efficient evaluation over inverted indexes. In fact, we employ a multiplicative factorization to model the interaction effects for several (page, ad) regions in a parsimonious way that facilitates fast look-up of ads at run time. Through large scale experiments, we convincingly demonstrate the advantage of combining relevance with click feedback. In fact, we achieve a 25% lift in precision for a recall value of 10% relative to a pure relevance based model in our experiments.

In the future, we want to enrich our models to discover interactions that are caused due to word synonyms both using a supervised and unsupervised approach. We are currently experimenting with a wide range of feature selection and model fitting scheme for the logistic regression proposed



(a) Page Main Effects

(b) Ad Main Effects

(c) Page Ad Interaction Effects

Figure 7: ***Average estimated weight for page main effects, ad main effects and interaction effects. The distribution is obtained as product of parameter estimate from regression and the average tf score associated with a word in the training corpus.***

in the paper. We are also exploring the application of our method to other systems where there is implicit feedback in the form of clicks, such as web search and search advertising.

7. REFERENCES

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

[2] 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.

[3] A. Z. Broder, M. Fontoura, V. Josifovski, and L. Riedel. A semantic approach to contextual advertising. In *SIGIR*, pages 559–566, 2007.

[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] C.Lin, R.C.Weng, and S.S.Keerthi. Trust region newton methods for large-scale logistic regression. In *International Conference on machine learning*, 2007.

[6] C.R.Rao. *Linear Statistical Inference and its Applications*. Wiley-Interscience, 2002.

[7] D.C.Liu and J.Nocedal. On the limited memory bfgs method for large scale optimization. *Mathmematical Programming*, 45:503–528, 1989.

[8] S. Derksen and H. J. Keselman. Backward, forward and stepwise automated subset selection algorithms: Frequency of obtaining authentic and noise variables. *British Journal of Mathematical and Statistical Psychology*, 45:265–282, 1992.

[9] Online ad spending to total $19.5 billion in 2007. eMarketer, February 2007. Available from `http://www.emarketer.com/Article.aspx?id=1004635`.

[10] A. Foundation. Apache hadoop project. In *lucene.apache.org/hadoop.*

[11] G.King and L.Zeng. Logistic regression in rare events data. *Political Analysis*, 9:137–162, 2001.

[12] J.Dean and S.Ghemawat. Mapreduce:simplified data processing on large clusters. In *Sixth Symposium on Operating System Design and Implementation*, pages 137–150, 2004.

[13] A. Lacerda, M. Cristo, M. A. 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.

[14] P. McCullagh and J.A.Nelder. *Generalized Linear Models*. Chapman and Hall, 1989.

[15] M.J.Silvapulle. On the existence of maximum likelihood estimates for the binomial response models. *Journal of the Royal Statistical Society, Series B*, 43:310–313, 1981.

[16] P.Komarek and A.W.Moore. Making logistic regression a core data mining tool with tr-irls. In *International Conference on Data Mining*, pages 685–688, 2005.

[17] M. Regelson and D. Fain. Predicting click-through rate using keyword clusters. In *In Proc. of the Second Workshop on Sponsored Search Auctions*, 2006.

[18] 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.

[19] M. Richardson, E. Dominowska, and R. Ragno. Predicting clicks: estimating the click-through rate for new ads. In *WWW*, pages 521–530, 2007.

[20] S.D.Pietra, V.D.Pietra, and J.Lafferty. Inducing features of random fields. *IEEE PAMI*, 19:380–393, 1997.

[21] 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.

[22] 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.