Filling Context-Ad Vocabulary Gaps with Click Logs

Yukihiro Tagami
Yahoo Japan Corporation
Tokyo, Japan
yutagami@yahoo-corp.jp

Toru Hotta
Yahoo Japan Corporation
Tokyo, Japan
thotta@yahoo-corp.jp

Yusuke Tanaka
Yahoo Japan Corporation
Tokyo, Japan
yuustana@yahoo-corp.jp

Shingo Ono
Yahoo Japan Corporation
Tokyo, Japan
shiono@yahoo-corp.jp

Koji Tsukamoto
Yahoo Japan Corporation
Tokyo, Japan
kotsukam@yahoo-corp.jp

Akira Tajima
Yahoo Japan Corporation
Tokyo, Japan
atajima@yahoo-corp.jp

ABSTRACT

Contextual advertising is a form of textual advertising usually displayed on third party Web pages. One of the main problems with contextual advertising is determining how to select ads that are relevant to the page content and/or the user information in order to achieve both effective advertising and a positive user experience. Typically, the relevance of an ad to page content is indicated by a tf-idf score that measures the word overlap between the page and the ad content, so this problem is transformed into a similarity search in a vector space. However, such an approach is not useful if the vocabulary used on the page is expected to be different from that in the ad. There have been studies proposing the use of semantic categories or hidden classes to overcome this problem. With these approaches it is necessary to expand the ad retrieval system or build new index to handle the categories or classes, and it is not always easy to maintain the number of categories and classes required for business needs. In this work, we propose a translation method that learns the mapping of the contextual information to the textual features of ads by using past click data. The contextual information includes the user's demographic information and behavioral information as well as page content information. The proposed method is able to retrieve more preferable ads while maintaining the sparsity of the inverted index and the performance of the ad retrieval system. In addition, it is easy to implement and there is no need to modify an existing ad retrieval system. We evaluated this approach offline on a data set based on logs from an ad network. Our method achieved better results than existing methods. We also applied our approach with a real ad serving system and compared the online performance using A/B testing. Our approach achieved an improvement over the existing production system.

Categories and Subject Descriptors

H.3.5 [**Information Storage and Retrieval**]: Online Information Services—*Commercial services*; I.2.6 [**Artificial Intelligence**]: Learning

Keywords

Contextual advertising, Learning-to-rank, Click feedback, Modeling.

1. INTRODUCTION

Online advertising is a key component supporting today's Internet ecosystem and is growing into a multi-billion dollar industry. Many different types of advertising are used: sponsored search advertising, contextual advertising, display advertising, real-time bidding auctions, and more [28]. In this paper, we focus on contextual advertising, which consists of short text messages that are usually displayed on third party Web pages such as news sites or blogs. The advertiser is primarily interested in targeting relevant users, and the publisher is concerned with keeping the user experience pleasant. To satisfy these two objectives, an adnetworking service selects ads that are relevant to the page content and/or the user information. In this paper, we focus on increasing the click-through rate (CTR), as this metric directly relates to the user experience, publisher revenue, and advertising effectiveness objectives.

The relevance of an ad to page content is typically determined using a tf-idf score that measures the word overlap between the page content and the ad content. This task is therefore regarded as a similarity search using an inverted index. This is an effective technique when the expected word overlap rate is high, but it falters if the vocabulary used on the page is different from the vocabulary used in the ad. For example, an ad for "SIM-free smartphones" would be related to a Web page comparing of Mobile Virtual Network Operator (MVNO) services, but the word overlap might not be very high. Another example could be an ad for "HTC One" and a page about "New Nexus 7."

Some previous studies have used a semantic taxonomy in the matching function [4] or introduced a page-ad probability model with hidden classes [21]. However, in these approaches, it is necessary to expand the ad retrieval system or build new index to handle the categories or classes. In addition, a review of the number and hierarchical structure of categories or a re-creation of clusters is periodically re-

KDD'14, August 24–27, 2014, New York, NY, USA.
Copyright is held by the owner/author(s). Publication rights licensed to ACM.
ACM 978-1-4503-2956-9/14/08 ...$15.00.
http://dx.doi.org/10.1145/2623330.2623334.

quired in the operation of the ad serving system, and these tasks are not always easy to perform.

To overcome the above problems, we have developed an approach that calculates a matching score between two term vectors using an inverted index and does not require modification of an ordinary ad retrieval system. In other words, this approach translates ad request information into the textual space of ads. With this translation table, the feature vector of ad requests is transformed into the input term vector of the ad retrieval system. The process is illustrated in Fig. 1.

This translation table will become very large because the two spaces are typically quite large. We can efficiently learn the translation table from past click data with low-rank approximation of the matrix. However, even if the learning can be done efficiently, it is necessary to make the transformed term vector sparse because the performance of the ad retrieval system gets progressively worse in accordance with the number of nonzero values in the input vector. Therefore, we first select ad features related to each query feature according to our metric based on the past CTR, and then we learn the translation matrix. In other words, we choose a subset of the matrix elements and learn only the weights with past click logs.

We reported an overview of this method and preliminary experimental results in our work-in-progress paper [23]. In the current paper, we describe the details of the improved method and report the extensive evaluations. Our main contributions are as follows.

- We propose a method of translating contextual information into the textual features of ads by using past click data.
- Our approach is easy to implement and maintain because there is no need to modify the existing ad retrieval system.
- We evaluated our approach using a real-world data set from an ad network and obtained better results than existing methods.
- We applied our approach to a real ad serving system and achieved an improvement over the existing production system.

The rest of this paper is organized as follows. Section 2 provides a general overview of the contextual advertising system. Section 3 presents our method of translating query information into ad term vector space. Section 4 details the experimental setup and results. We conclude our paper in Section 5 by summarizing our findings and making a brief mention of future work.

2. OVERVIEW AND RELATED WORKS

This section provides a general overview of contextual advertising and related studies.

2.1 Overview of Contextual Advertising

There are four players in contextual advertising: the publisher, the advertiser, the ad network, and the user. The publisher owns Web pages and reserves some space on these pages for ads. An ad network supplies the ads that an advertiser provides. In the common pay-per-click model, which is the prevalent pricing model for contextual advertising, the advertiser pays the Web publisher and the ad network a fee only if a user clicks on their advertisement and visits their website. Thus, the expected revenue from displaying each ad is a function of both the bid price and the CTR. The bid price is the cost that the advertiser agrees to pay per click, so the advertising system already knows this. In contrast, the CTR for each ad can vary significantly depending on a variety of factors ranging from the Web page to the user. Consequently, one of the main problems in contextual advertising is determining how to accurately predict CTR and efficiently retrieve ads with a high CTR from an ad corpus.

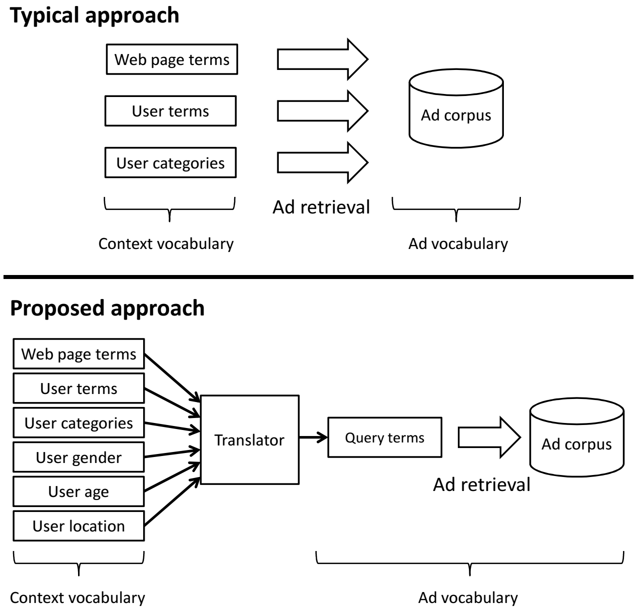

Figure 1: Typical approach (top) vs. our proposed approach (bottom). Proposed method is a translation of request features into the input term vector of an ad retrieval system.

2.2 Related Works

Some previous works have focused on developing methods to match ads to pages, since ads that are related to the page content are more likely than generic ads to provide a better user experience. That in turn increases the probability of users clicking on the ads. In these studies, the problem of matching ads with pages is transformed into a similarity search in a vector space. The relevance of an ad to page content is indicated by a tf-idf score that measures the word overlap between the page content and the ad content. Chakrabarti et al. [6] and Karimzadehgan et al. [16] introduced methods to learn the weights of each word in a page and an ad using HTML tags and ad sections. They use an inverted index to efficiently retrieve top-K items from ad corpora.

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 on the pages. Various approaches have been proposed to overcome this problem. Broder et al. [4] used a 6000-node semantic taxonomy in the matching function between pages and ads. In addition, Ratna-

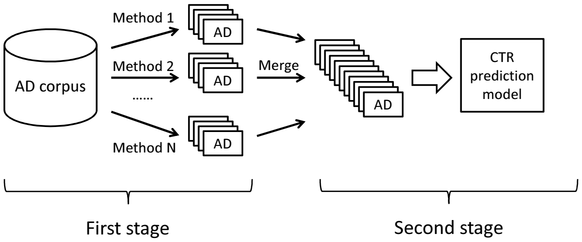

Figure 2: Two-stage approach in the ad serving system. Ads are retrieved by multiple methods in the first stage. The ads are merged and passed on to the second stage for CTR prediction.

parkhi [21] introduced a page-ad probability model in which semantic relationships between page terms and ad terms are modeled with hidden classes. Yih and Jiang [27] proposed an approach to map the original term vectors to a "concept space" so that semantically close words would be captured by the same concept. Wang et al. [26] formulated and tackled the problem of relevance learning for online targeting in heterogeneous social networks. They inferred user interests and ad concepts from heterogeneous sources and links, and developed a user-ad relevance feature based on weighted matching between any pair of concept classes. Murdock et al. [20] applied machine translation techniques to improve the matching between pages and ads.

Joshi et al. [15] presented a method to leverage user information including a user's demographic information (e.g., age, gender, and location) and behavioral information (e.g., the user's recent search history, page visits, and ad clicks) in a content match advertising setting. They mapped the non-textual user features to the textual space of ads.

In the cases where ad relevance cannot easily be gleaned from the page text alone, the "clickable terms" approach has been proposed [13]. This approach involves matching a Web site directly with a set of ad side terms, independent of the page content.

Another line of research attempts to predict the CTR of ads. These studies are not only related to contextual advertising but also to sponsored search advertising because both typically employ the pay-per-click model. Predictions of CTR for ads are generally based on a statistical model trained by using past click data. Examples of such models include logistic regression [7, 8, 19], probit regression [12], and boosted trees [9, 25]. The accuracy of the model depends greatly on the design of the features. Cheng and Cantú-Paz [7] presented a framework for the personalization of click models. These authors developed user-specific and demographic-based features that reflect the click behavior of individuals and groups. These features are based on observations of the search and click behaviors of a large number of users of a commercial search engine. Other recent works [1, 17, 22] have proposed models to estimate conversion rates (CVR).

2.3 Our Ad Serving System

Although machine-learned CTR prediction models are more accurate, as described in Section 2.2, evaluating all the items in an ad corpus is too time-consuming if we want to ensure that Web page loads quickly [2]. Our ad serving system therefore adopts a two-stage approach similar to some other studies [2, 6]. The first stage retrieves top-K items from an ad corpus using an inverted index. The second stage selects the desired top-k using brute force CTR prediction on the K retrieved ads ($k \ll K$). In the first stage, ads are independently retrieved by multiple methods in parallel. The retrieved ads in the first stage are merged and passed on to the second stage for CTR prediction. Ads are ranked and displayed on the basis of the predicted CTR and the bid price. This process is illustrated in Fig. 2. We refer to the "ad retrieval system" as a part of the first stage and focus on the part. We have used a heavily tuned search engine that employs the WAND algorithm [5, 11, 10].

In this paper, we propose a method to efficiently retrieve ads with high CTR using the search engine without any changes, even building new index. Unlike the method proposed by Agarwal and Gurevich [2] or approaches with semantic categories [4] or hidden classes [21], our method simply utilize the existing index used by typical similarity search in a vector space.

3. METHODS

In this section, we first define the matching function between a query and an ad and then describe our approach to mapping the query information to the textual space of ads.

3.1 Matching Function

We define a query feature vector as $\boldsymbol{q} = (q_1, \ldots, q_{D_q})^{\mathrm{T}}$ and an ad feature vector as $\boldsymbol{a} = (a_1, \ldots, a_{D_a})^{\mathrm{T}}$. $\boldsymbol{q}$ is a query feature vector of an ad request, which includes Web page and user information. We also define a general form of the matching function for $\boldsymbol{q}$ and $\boldsymbol{a}$ using translation matrix $\boldsymbol{W} = [w_{ij}]_{D_q \times D_a}$, as:

$$mscore(\boldsymbol{q}, \boldsymbol{a}) = \boldsymbol{q}^{\mathrm{T}}\boldsymbol{W}\boldsymbol{a} = \sum_{i=1}^{D_q}\sum_{j=1}^{D_a} w_{ij} q_i a_j, \tag{1}$$

where D_q is a dimension of query feature space and D_a is a dimension of ad feature space.

Equation (1) is a general form of the matching function. If D_a is the same as D_q and $\boldsymbol{W}$ is the identity matrix $\boldsymbol{I}$, score$(\boldsymbol{q}, \boldsymbol{a})$ turns out to be a simple dot product:

$$mscore(\boldsymbol{q}, \boldsymbol{a}) = \boldsymbol{q}^{\mathrm{T}}\boldsymbol{I}\boldsymbol{a} = \sum_{i=1}^{D_q} q_i a_i$$

Of course, the cosine similarity can be calculated if $\boldsymbol{q}$ and $\boldsymbol{a}$ are normalized. Also, in general cases, by multiplying the query feature vector $\boldsymbol{q}$ by the translation matrix $\boldsymbol{W}$ and using $\boldsymbol{q}' = \boldsymbol{W}^{\mathrm{T}}\boldsymbol{q}$ as an input vector, we calculate the score $mscore(\boldsymbol{q}, \boldsymbol{a}) = \boldsymbol{q}^{\mathrm{T}}\boldsymbol{W}\boldsymbol{a} = \boldsymbol{q}'^{\mathrm{T}}\boldsymbol{a}$ with an ordinary inverted index based ad retrieval system.

When categories or classes are used, translation matrix $\boldsymbol{W}$ is decomposed as:

$$\boldsymbol{W} = \boldsymbol{\Gamma}_q^{\mathrm{T}}\boldsymbol{W}_c\boldsymbol{\Gamma}_a,$$

where $\boldsymbol{\Gamma}_q \in \mathbb{R}^K \times \mathbb{R}^{D_q}$, $\boldsymbol{\Gamma}_a \in \mathbb{R}^K \times \mathbb{R}^{D_a}$, $\boldsymbol{W}_c \in \mathbb{R}^K \times \mathbb{R}^K$, and K is the number of categories or classes. Here, $\boldsymbol{\Gamma}_q$ and $\boldsymbol{\Gamma}_a$ are the respective matrices that convert query feature vector $\boldsymbol{q}$ and ad feature vector $\boldsymbol{a}$ into categories or classes vector $\boldsymbol{q}_c$ and $\boldsymbol{a}_c$. Thus, the matching function in this case is expressed as:

$$\begin{aligned} mscore(\boldsymbol{q}, \boldsymbol{a}) &= (\boldsymbol{\Gamma}_q \boldsymbol{q})^{\mathrm{T}} \boldsymbol{W}_c (\boldsymbol{\Gamma}_a \boldsymbol{a}) \\ &= \boldsymbol{q}_c^{\mathrm{T}} \boldsymbol{W}_c \boldsymbol{a}_c, \end{aligned}$$

where $\boldsymbol{q}_c = \boldsymbol{\Gamma}_q \boldsymbol{q}$ and $\boldsymbol{a}_c = \boldsymbol{\Gamma}_a \boldsymbol{a}$. This decomposition can also be viewed as a matrix factorization or low-rank approximation.

3.2 Learning a Translation Matrix

As described in Section 1, we need to learn the translation matrix $\boldsymbol{W}$ efficiently and make the transformed term vector sparse for efficient ad retrieval. The translation matrix $\boldsymbol{W}$ can be learned directly with a large hash table and L1 regularization, as Wang et al. did [26]. In this paper, we propose another approach, in which we can directly control the sparseness of the matrix by considering the performance of the ad retrieval system. We first select a subset of the matrix elements and then learn the corresponding w_{ij}.

We calculate the following score m_{ij} for each pair of q_i and a_j presented in the training data:

$$m_{ij} = \frac{ctr(q_i, a_j)}{\max(ctr(q_i), ctr(a_j))},$$

where $ctr(q_i)$ denotes the CTR when the query feature vector includes q_i and $ctr(a_j)$ denotes the CTR of ads that include feature a_j. Similarly, $ctr(q_i, a_j)$ represents the CTR of ads that include feature a_j when the query feature vector includes q_i. A large m_{ij} value means that ads that include feature a_j are more likely to be clicked when the query feature vector includes q_i. A set of the pairs that has a larger m_{ij} is selected:

$$P = \{(i, j) \mid m_{ij} > T\},$$

where T is a thresholding hyper-parameter. The number of non-zero elements in $\boldsymbol{W}$ decreases as a function of T. We use P and replace the matching function (1) as follows.

$$mscore(\boldsymbol{q}, \boldsymbol{a}) = \sum_{(i,j) \in P} w_{ij} q_i a_j$$

If we perform an ordinary feature selection using mutual information or L1 regularization, we obtain features that lead negative w_{ij}'s as well as positive w_{ij}'s. However, we retrieve top-K ads which have larger $mscore(\boldsymbol{q}, \boldsymbol{a})$ from an ad corpus as described in Section 2.3. We cannot retrieve ads that have higher $mscore(\boldsymbol{q}, \boldsymbol{a})$ even if we use the negative w_{ij}'s. Thus, we employ the above approach in the hope of selecting the features to lead the positive w_{ij}'s.

We learn the above w_{ij} by using past click data. A score proportional to the click-through rate (CTR) for $\boldsymbol{q}$ and $\boldsymbol{a}$ is therefore defined as having the following linear form:

$$\begin{aligned} score(\boldsymbol{q}, \boldsymbol{a}) &= mscore(\boldsymbol{q}, \boldsymbol{a}) + bscore(\boldsymbol{q}, \boldsymbol{a}) \\ &= \sum_{(i,j) \in P} w_{ij} q_i a_j + \boldsymbol{w}_{basic}^{\mathrm{T}} \boldsymbol{x}_{basic} \\ &= \boldsymbol{w}_{match}^{\mathrm{T}} \boldsymbol{x}_{match} + \boldsymbol{w}_{basic}^{\mathrm{T}} \boldsymbol{x}_{basic} \\ &= \boldsymbol{w}^{\mathrm{T}} \boldsymbol{x}, \end{aligned} \quad (2)$$

where $bscore(\boldsymbol{q}, \boldsymbol{a})$ is a basic score as $bscore(\boldsymbol{q}, \boldsymbol{a}) = \boldsymbol{w}_{basic}^{\mathrm{T}} \boldsymbol{x}_{basic}$. $\boldsymbol{x}_{basic}$ is a feature vector that includes features such as the ad's own clickability. $\boldsymbol{w}_{basic}$ is a weight vector corresponding to $\boldsymbol{x}_{basic}$. For $(i, j) \in P$, $\boldsymbol{w}_{match}$ and $\boldsymbol{x}_{match}$ are defined as $\boldsymbol{w}_{match} = [\cdots, w_{ij}, \cdots]^{\mathrm{T}}$ and $\boldsymbol{x}_{match} = [\cdots, q_i a_j, \cdots]^{\mathrm{T}}$. $\boldsymbol{w} = \left[\boldsymbol{w}_{match}^{\mathrm{T}}, \boldsymbol{w}_{basic}^{\mathrm{T}}\right]^{\mathrm{T}}$ and $\boldsymbol{x} = \left[\boldsymbol{x}_{match}^{\mathrm{T}}, \boldsymbol{x}_{basic}^{\mathrm{T}}\right]^{\mathrm{T}}$ are concatenated vectors. The reason for adding $bscore(\boldsymbol{q}, \boldsymbol{a})$ to $mscore(\boldsymbol{q}, \boldsymbol{a})$ is that the score proportional to CTR consists of not only a matching score between query and ad but also other factors such as the ad's own clickability and display position on the Web page.

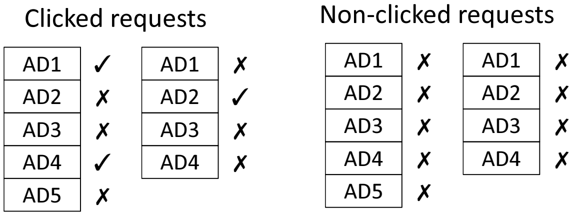

Figure 3: Examples of clicked requests and non-clicked requests. Ticks denote clicked impressions and x's represent non-clicked impressions.

As described in Section 2.3, our focus is improving the first stage of the two-staged ad serving system. We learn w_{ij} with click logs instead of simply using m_{ij}, as we expect that first retrieving top-K ads based on a score approximating the second stage score leads to better top-k results at the second stage.

We then describe how to utilize click logs to learn weights. In the contextual advertising setting, a Web publisher typically requests some ads simultaneously because more than one ad is displayed on a page at the same time. In other words, one ad request r in the logs includes $N^{(r)}$ impressions of ads:

$$(\boldsymbol{q}^{(r)}, \boldsymbol{a}_1^{(r)}, y_1^{(r)}), (\boldsymbol{q}^{(r)}, \boldsymbol{a}_2^{(r)}, y_2^{(r)}), \ldots, (\boldsymbol{q}^{(r)}, \boldsymbol{a}_{N^{(r)}}^{(r)}, y_{N^{(r)}}^{(r)})$$

Each impression of an ad consists of a tuple $(\boldsymbol{q}^{(r)}, \boldsymbol{a}_i^{(r)}, y_i^{(r)})$. Here, $\boldsymbol{q}^{(r)}$ represents the query feature vector of the request r and $\boldsymbol{a}_i^{(r)}$ represents the ad feature vector of the ith impression in request r. The output variable $y_i^{(r)} = 1$ if a user clicked the ad and $y_i^{(r)} = 0$ if not.

As described in our previous work [24], we focus on a kind of ad request and apply the learning-to-rank approach to learn the weights.

There are two kinds of ad requests in data R:

$$\begin{aligned} R^+ &= \{r \mid \exists i (y_i^{(r)} = 1)\} \\ R^- &= \{r \mid \forall i (y_i^{(r)} = 0)\} \end{aligned}$$

R^+ denotes a set of ad requests that includes at least one clicked impression; hence we refer to R^+ as **clicked requests**. R^- is called **non-clicked requests** since no ad requests in R^- include clicked impressions. Of course, $R = R^+ \cup R^-$ and $R^+ \cap R^- = \emptyset$. Examples of these two kinds of ad request are shown in Fig. 3.

We regard ad impressions in an ad request as documents related to a query in an information retrieval context and apply the learning-to-rank approach to learn the weights. We make pairwise preferences from each clicked request $r \in R^+$.

$$\{(\boldsymbol{a}_i^{(r)}, \boldsymbol{a}_j^{(r)}) \mid \forall i, j (y_i^{(r)} = 1 \wedge y_j^{(r)} = 0)\}$$

This process is illustrated in Fig. 4.

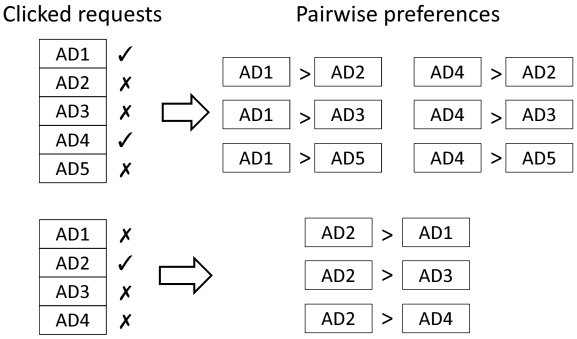

Figure 4: Making pairwise preferences from clicked requests.

The preference $(\boldsymbol{a}_i^{(r)}, \boldsymbol{a}_j^{(r)})$ indicates that a score proportional to the CTR of $\boldsymbol{a}_i^{(r)}$ for $\boldsymbol{q}^{(r)}$ is expected to be higher than that of $\boldsymbol{a}_j^{(r)}$. We represent the above preference using $score(\boldsymbol{q}, \boldsymbol{a})$ and transform using (2) as follows:

$$\begin{aligned}
& score(\boldsymbol{q}^{(r)}, \boldsymbol{a}_i^{(r)}) > score(\boldsymbol{q}^{(r)}, \boldsymbol{a}_j^{(r)}) \\
\Leftrightarrow\ & score(\boldsymbol{q}^{(r)}, \boldsymbol{a}_i^{(r)}) - score(\boldsymbol{q}^{(r)}, \boldsymbol{a}_j^{(r)}) > 0 \\
\Leftrightarrow\ & \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i^{(r)} - \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_j^{(r)} > 0 \\
\Leftrightarrow\ & \boldsymbol{w}^{\mathrm{T}}(\boldsymbol{x}_i^{(r)} - \boldsymbol{x}_j^{(r)}) > 0
\end{aligned}$$

Using the squared hinge loss, we define a pairwise loss function $L(\boldsymbol{w})$ like RankSVM [14], as follows:

$$L(\boldsymbol{w}) = \sum_{r \in R^+} \sum_{i:y_i^{(r)}=1} \sum_{j:y_j^{(r)}=0} \max(0, 1 - \boldsymbol{w}^{\mathrm{T}}(\boldsymbol{x}_i^{(r)} - \boldsymbol{x}_j^{(r)}))^2$$

We add a regularization term and seek the weight vector $\hat{\boldsymbol{w}}$ that minimizes the following optimization problem:

$$\hat{\boldsymbol{w}} = \arg\min_{\boldsymbol{w}} \frac{1}{2}\|\boldsymbol{w}\|_2^2 + C \cdot L(\boldsymbol{w}),$$

where $C \geq 0$ is a penalty parameter. The translation matrix $\boldsymbol{W}$ is restored from the weight vector $\hat{\boldsymbol{w}}_{match}$, where $\hat{\boldsymbol{w}} = \left[\hat{\boldsymbol{w}}_{match}^{\mathrm{T}}, \hat{\boldsymbol{w}}_{basic}^{\mathrm{T}}\right]^{\mathrm{T}}$. Note that we set $w_{ij} = 0$ for $(i, j) \notin P$.

We conducted preliminary experiments using hinge loss and logistic loss in addition to the above squared hinge loss. We decided to use the squared hinge loss as it was found to have a favorable balance between accuracy and training time.

3.3 Retrieval from AD Corpus and Implementation

With the learned matrix $\mathbf{W}$, the query feature vector $\boldsymbol{q}$ is transformed into the input term vector of the ad retrieval system for each ad request.

$$\boldsymbol{q}_{input} = \boldsymbol{W}^{\mathrm{T}}\boldsymbol{q}$$

Our proposed method only require this transformation. Thus it is easy to implement and maintain because there is no need to modify the existing inverted index or add new index.

This input term vector includes some non-zero values, which are proportional to the number of non-zero values in the query feature vector. However, the performance of the ad retrieval system declines in accordance with the number of non-zero values in the input vector. Therefore, we need to limit the number of these values with hyper-parameter M:

$$\|\boldsymbol{q}_{input}\|_0 \leq M$$

$\|\boldsymbol{q}_{input}\|_0$ is the L_0-norm of $\boldsymbol{q}_{input}$, which is the number of its non-zero elements. We simply choose top-M elements, which have larger values.

4. EXPERIMENT

This section describes offline and online evaluations. Due to business confidentiality, we report only relative performance when showing experimental results.

4.1 Offline Evaluation

In this section, we first describe experimental settings such as data sets, features, models, and evaluation metric. Next, we compare our approach with existing methods and present model performances when changing the hyper-parameters T and M.

4.1.1 Data Sets and Features

We compare the models using data sampled from an ad network for a period of eight weeks. Data from the first six weeks are used as a training set, data from the fifth week are used as a validation set, and data from the last week are treated as a testing set. This ad network is for the Japanese market[1], so all ads and pages are written in Japanese, with a few exceptions.

As described in Section 3.2, each sample of the data sets is an impression of an ad and consists of a tuple $(\boldsymbol{q}^{(r)}, \boldsymbol{a}_i^{(r)}, y_i^{(r)})$. The output variable $y_i^{(r)} = 1$ if a user clicks the ad and 0 if not. The query features $\boldsymbol{q}^{(r)}$ include Web page and user information. The Web page features are extracted terms. These terms are scored on the basis of their position on the page and HTML tags. Some terms are chosen by the score. The user features are terms and categories in which the user is interested, as well as gender, age, and location. The user's gender falls into three classes: male, female, and unknown. Similarly, the user's age is categorized into thirteen groups. As in the study by Aly et al. [3], these terms and categories are extracted from user behavior events such as page visits, search queries, and ad clicks. These categories are similar to the hierarchical taxonomy in the work of Broder et al. [4]. Number of the categories is about 900. We simply use textual features as the ad features $\boldsymbol{a}_i^{(r)}$, which are tf-idf weighted terms based on the title and description in this paper. These features are summarized in Table 1.

The basic features $\boldsymbol{x}_{basic}$ described in Section 3.2 include the display position on the Web page and the historical CTR of the ad and advertiser. These features are also summarized in Table 1.

We chose eight diverse Web sites, including news, blogs, question-and-answer, finance, sports, weather, and travel sites. The models we evaluate are constructed with respect to each Web site, since the Web pages and the users that visit them are different. The data statistics for each Web site are summarized in Table 2. The number of clicked requests $\left|R^+\right|$ and the average number of impressions per

[1] `http://promotionalads.yahoo.co.jp/service/ydn/index.html`

Table 1: Summary of features

Feature type	Source	Details
Query features $\boldsymbol{q}$	Web page	Terms extracted from Web page
	User	Terms extracted from behavioral events, categories based on behavioral events, gender, age, location
Ad features $\boldsymbol{a}$	Ad	Tf-idf weighted terms
Basic features $\boldsymbol{x}_{basic}$	Past click log	Historical CTR of ad and advertiser, display position on the Web page

Table 2: Data statistics for Web sites used in evaluation.

Web site	Type	$\lvert R^+ \rvert$	$\overline{N^{(r)}}$	$\overline{\#clicks}$
A	training	711,539	6.97	1.03
	validation	142,649	7.03	1.03
	testing	119,464	7.11	1.02
B	training	2,676,577	4.99	1.03
	validation	429,760	4.99	1.03
	testing	356,578	4.99	1.01
C	training	1,648,118	4.64	1.02
	validation	137,646	3.09	1.02
	testing	134,168	3.15	1.01
D	training	919,870	4.11	1.01
	validation	92,547	4.05	1.01
	testing	81,077	4.07	1.00
E	training	905,842	4.07	1.01
	validation	217,165	4.94	1.01
	testing	169,627	4.94	1.01
F	training	153,849	5.00	1.04
	validation	23,836	5.00	1.04
	testing	17,879	5.00	1.02
G	training	297,814	8.13	1.03
	validation	53,306	8.10	1.02
	testing	42,098	8.05	1.02
H	training	4,644,350	4.48	1.02
	validation	780,037	4.54	1.02
	testing	498,407	4.15	1.01

clicked request $\overline{N^{(r)}}$ changed over the six weeks, due to seasonal trends, changes in the budget of the advertisers, and actions carried out by publishers to achieve sales targets. $\overline{\#clicks}$, which is the average number of clicked impressions per clicked request, is approximately 1.

4.1.2 Existing Methods and Evaluation Metric

We compared the proposed method with three existing in production methods: *existing 1*, *2*, and *3*. *Existing 1* utilizes only terms extracted from Web page for ad retrieval and *existing 2* and *3* use terms and categories based on user's behavioral events respectively. Please also refer Section 4.1.1 and Table 1.

In the comparison, we use the following $score_{existing}(\boldsymbol{q}, \boldsymbol{a})$ instead of $score(\boldsymbol{q}, \boldsymbol{a})$ in Equation (2):

$$score_{existing}(\boldsymbol{q}, \boldsymbol{a}) = w \cdot escore(\boldsymbol{q}, \boldsymbol{a}) + bscore(\boldsymbol{q}, \boldsymbol{a}),$$

where $escore(\boldsymbol{q}, \boldsymbol{a})$ is a matching score calculated by an existing method. Each existing method has a different $escore(\boldsymbol{q}, \boldsymbol{a})$.

As described in Section 3.3, we need to limit the number of query terms because of the performance of the ad retrieval system. In the experiment, we carried out our evaluation by changing the value of M. Thus, we rewrite the scoring function for the prediction as:

$$t\text{-}score(\boldsymbol{q}, \boldsymbol{a}) = t\text{-}mscore(\boldsymbol{q}, \boldsymbol{a}) + bscore(\boldsymbol{q}, \boldsymbol{a}), \qquad (3)$$

where $t\text{-}mscore(\boldsymbol{q}, \boldsymbol{a})$ is a matching function using truncated $\boldsymbol{q}_{input} = \boldsymbol{W}^{\mathrm{T}}\boldsymbol{q}$. We change M and truncate the query term vector $\boldsymbol{q}_{input}$ during the evaluation, not during training. This means the same translation matrix $\boldsymbol{W}$ is used. Note that this evaluation does not reflect the actual online setting very well when the value of M is limited. We use $t\text{-}score(\boldsymbol{q}, \boldsymbol{a})$ in the offline evaluation, although ads are retrieved by $t\text{-}mscore(\boldsymbol{q}, \boldsymbol{a})$ from an ad corpus in the actual online setting. Because of $bscore(\boldsymbol{q}, \boldsymbol{a})$, the order by $t\text{-}mscore(\boldsymbol{q}, \boldsymbol{a})$ and $t\text{-}score(\boldsymbol{q}, \boldsymbol{a})$ is not the same for some ad requests in the testing set.

We evaluated the performance of the model by using mean average precision (MAP) [18]:

$$\begin{aligned} MAP &= \frac{1}{|R^+|} \sum_{r \in R^+} AP^{(r)} \\ AP^{(r)} &= \frac{\sum_{k=1}^{N^{(r)}} P_k^{(r)} y_{\pi^{(r)}(k)}^{(r)}}{\sum_{k=1}^{N^{(r)}} y_{\pi^{(r)}(k)}^{(r)}} \\ P_k^{(r)} &= \frac{\sum_{l=1}^{k} y_{\pi^{(r)}(l)}^{(r)}}{k}, \end{aligned}$$

where $AP^{(r)}$ is the averaged precision over all relevant documents for request r, and $P_k^{(r)}$ is the precision up to rank position k. Here, $\pi^{(r)}(k) = i$ means that the ith impression ranks in the kth position by the predicted score $score(\boldsymbol{q}^{(r)}, \boldsymbol{a}_j^{(r)})$.

We normalize the scores of the method by a basic model that uses only $bscore(\boldsymbol{q}, \boldsymbol{a})$ during both the training and evaluation. All values of metrics in this paper are transformed by

$$\Delta MAP = \left(\frac{MAP}{MAP_{basic}} - 1\right) \times 100.$$

Note that this MAP_{basic} is reasonably high because the display position included in $\boldsymbol{x}_{basic}$ is a very beneficial feature.

4.1.3 Results

We first evaluated our approach when changing the thresholding hyper-parameter T. As described in Section 3.2, the number of non-zero elements in $\boldsymbol{W}$ decreases as a function of T. This means that model performance is expected to improve with decreasing T. In this setting, M is not limited during the evaluation. The experimental results are summarized in Table 3. The **bold** elements indicate the best performance of the methods. Our proposed method achieved an improvement over the existing methods. As expected, MAPs are improved with decreasing T. For Web sites B and H, which has a lot of training data, ΔMAP is larger than other Web sites and the impact of changes in T is relatively small. For Web site A, the MAPs of *existing 1* are higher than the scores when $T = 0.20$ and 0.15. The impact of changes in T is large. This result indicates that the model trained with more data achieves larger and robust improvement. In comparing the existing methods, there are strong and weak points for each Web site. *Existing 1* was superior to the other two methods for Web sites A, E, and F. Conversely, *existing 2* achieved better results for Web sites B, C, D, G, and H. What this means is that the importance of the features used to retrieve ads is significantly different depending on the Web site. In contrast with these existing methods, our proposed method uses both Web page and user information for ad retrieval, which is why it had the better results.

Next, we investigated the model performance of each T when changing M. As described in Section 4.1.2, we change M and truncate the query term vector $\boldsymbol{q}_{input}$ during the evaluation, not during the training. The experimental results are shown in Fig. 5. As expected, the MAPs decay in response to a decrease of M. One might think that our proposed method is quite a bit worse than the basic model in situations where the ΔMAP is a negative value, such as $T = 0.05$ and $M = 50$ on Web site H. However, such offline evaluation results do not reflect the actual online performance very well because of the difference between $t\text{-}score(\boldsymbol{q}, \boldsymbol{a})$ and $t\text{-}mscore(\boldsymbol{q}, \boldsymbol{a})$, as described in Section 4.1.2. In the next section, the comparison of the existing methods and our proposed method when M is limited is carried over to the online evaluation. Here, we claim that M is first determined by the performance of the ad retrieval system and that T then needs to be tuned for each Web site in a real ad serving setting.

Tables 4 and 5 are examples of the mapping tables used on Web site B for user terms and categories, respectively. As expected, user terms are translated into the same term and related terms. In addition, the weight of the same term is larger than that of related terms in almost all cases. Similarly, user categories are translated into related terms.

4.2 Online Evaluation

The data sets used in the offline evaluation are based on past click data, which are results of ad serving by an existing system. Therefore, the results of offline evaluation might not reflect actual online performance because ads retrieved by our approach in the online setting are possibly different from those in the logs.

To measure the online performance, we applied our approach to a real ad serving system. This ad serving system adopts a two-stage approach, as described in Section 2.3 and shown in Fig. 2. We added the proposed method to the first stage and compared the online performance by conducting A/B testing. Hyper-parameters are set as ($T = 0.20, M = 20$). For a fair comparison, the total number of ads retrieved in the first stage is set to be the same because CTR can be higher even if the number of ads just increases. The CTR prediction model used for each version in the second stage was also the same. This prediction model was a statistical model trained by using the past click data [24]. We ran the online test over 1-week period in November 2013 for each Web site.

We use three metrics for the online test: CTR, cost per click (CPC), and revenue per request (RPR). These metrics are defined as follows:

$$\begin{aligned} CTR &= \frac{\#clicks}{|R|} \\ CPC &= \frac{revenue}{\#clicks} \\ RPR &= \frac{revenue}{|R|}, \end{aligned}$$

where $|R|$ denotes the number of ad requests. Revenue is the total amount of the fee that advertisers paid.

The experimental results are summarized in Table 6. These percentages also represent the relative gain. The CTR was improved for all Web sites expect site A. We simply set hyper-parameter as ($T = 0.20, M = 20$) for all Web sites although the result of $T = 0.20$ is worse than the result of *existing 1* and *2* on Web site A in Table 3. Thus, this result is reasonable and indicates that hyper-parameters need to be tuned for each Web site. For Web sites B and H, which has a lot of training data, the improvement of CTR is larger than other Web sites as well as offline test. We performed chi-squared test on the CTR results. The results of the Web sites A, B, D, E, and H are statistically significant at the 5% level (p-value < 0.05). The RPR was also improved for all Web sites expect A, whereas the CPC was decreased for Web sites B, E, G, and H. This drop in CPC is usually favored by the advertisers. In this online testing, ads were ranked and displayed by considering revenues. Ads retrieved by our proposed method had a high CTR and relatively low bid price, which is why the Web sites had the result. As described in Section 1, we focus on increasing CTR in this paper. Consequently, our proposed method improve revenue. This result indicates that our proposed method achieved an improvement in the online setting as well as the offline setting.

5. CONCLUSION AND FUTURE WORK

Contextual advertising is a form of textual advertising usually displayed on third party Web pages. Because of the need to achieve both effective advertising and a positive user experience, one of the main problems with contextual advertising is determining how to select ads that are relevant to the page content and/or the user information. In this paper, we introduced a translation method that learns a mapping of contextual information to the textual features of ads. The contextual information includes the user's demographic and behavioral information as well as Web page content information. Our proposed method only require the transformation of the context feature vector with a learned matrix into the input vector of the ad retrieval system. So it is easy to implement and there is no need to modify the existing inverted index or add new index. We evaluated this approach offline on a real-world data set from an ad network and obtained better results compared to existing methods. We also applied our approach with a real ad serving system and achieved improvement over the existing production system.

Table 3: Experimental results. Values are ΔMAP. The bold elements indicate the best performance of the methods.

Method	Web site							
	A	B	C	D	E	F	G	H
Existing 1	+1.90%	+0.01%−	0.04%	+0.01%	+0.87%	+1.22%	+0.31%	+0.00%
Existing 2	+0.85%	+2.71%	+0.53%	+0.24%	+0.85%	+0.59%	+0.98%	+1.00%
Existing 3	+0.04%	+0.37%	+0.04%	+0.03%	+0.02%	+0.38%	+0.01%	+0.11%
$T = 0.20$	+0.60%	+6.18%	+1.42%	+0.50%	+1.70%	+1.66%	+1.73%	+2.60%
$T = 0.15$	+1.16%	+6.60%	+1.50%	+0.60%	+1.79%	+2.14%	+1.86%	+2.90%
$T = 0.10$	+2.97%	+6.83%	**+1.55%**	+0.64%	**+1.84%**	+2.32%	+2.00%	+3.13%
$T = 0.05$	**+3.56%**	**+6.93%**	+1.54%	**+0.68%**	+1.81%	**+2.56%**	**+2.12%**	**+3.24%**

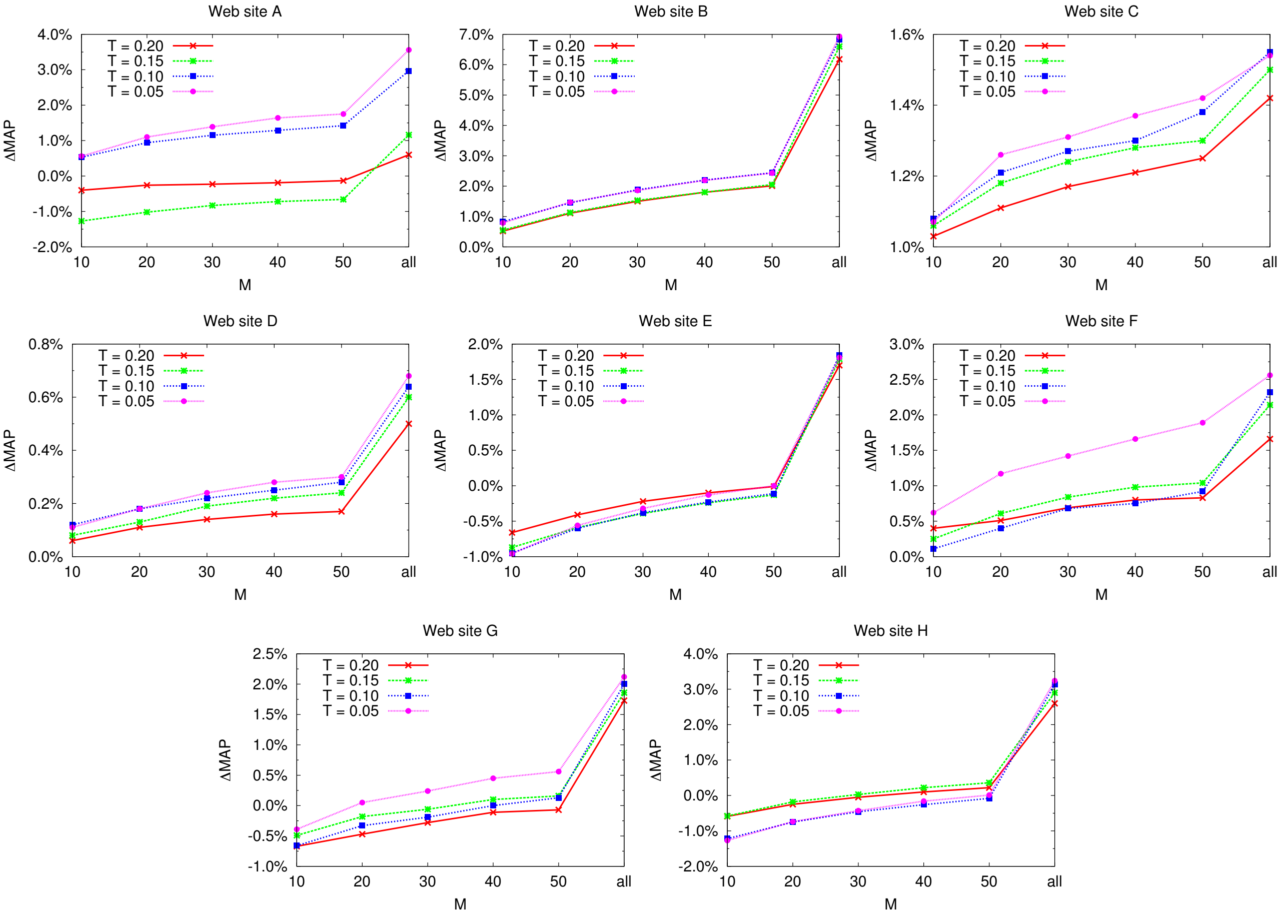

Figure 5: Experimental results when changing M.

Our future work will take three directions. First, we want to study the use of various types of context features such as ad request time and weather. Second, we plan to investigate sophisticated regularization term in objective function for making the weight matrix sparse. Finally, distributed online learning of the translation matrix is an interesting challenge that we wish to explore.

6. ACKNOWLEDGMENTS

We would like to thank our colleagues for their assistance with data collection, model evaluation, and many insightful discussions.

7. REFERENCES

[1] D. Agarwal, R. Agrawal, R. Khanna, and N. Kota. Estimating rates of rare events with multiple hierarchies through scalable log-linear models. In *Proceedings of the 16th ACM SIGKDD international*

Table 4: Example of Web site B's mapping table for user terms.

User term	Translated term	Weight
iPhone	iPhone	0.2114
	ケース (case)	0.1534
	iPad	0.0868
プリウス (Toyota Prius)	プリウス (Toyota Purius)	0.2600
	燃費 (mileage)	0.0732
	HV (Hybrid Vehicle)	0.0607
歯科 (dentistry)	歯科 (dentistry)	0.3297
	歯科医師 (dentist)	0.1892
	インプラント (implant)	0.1035
毛穴 (pores)	毛穴 (pores)	0.2319
	洗顔 (face washing)	0.1001
	化粧品 (cosmetics)	0.0663
温泉 (hot spring)	温泉 (hot spring)	0.1730
	旅館 (Japanese inn)	0.1272
	露天風呂 (outdoor hot spring)	0.0809
カーナビ (car navigation system)	カーナビ (car navigation system)	0.1229
	トヨタ (Toyota)	0.0906
	ホンダ (Honda)	0.0720

Table 5: Example of Web site B's mapping table user interest category.

User category	Translated term	Weight
Automotive/Domestic/Toyota	クラウン (Toyota Crown)	0.2605
	トヨタプリウス (Toyota Prius)	0.2171
	ランクル (Toyota Land Cruiser)	0.2053
Health Pharma/Adult Disease/Hypertensive Disease	血圧 (blood pressure)	0.1784
	高血圧 (high blood pressure)	0.1196
	食事法 (diet)	0.0531
Travel and Transportation/Overseas/Europe	海外 (overseas)	0.1181
	ヨーロッパ (Europe)	0.1168
	海外旅行 (foreign travel)	0.0868
Miscellaneous/Sex and Romance/Personals	婚活 (marriage hunting)	0.1100
	出会い (matchmaking)	0.0742
	カップル (couple)	0.0546
Life Stage/Wedding	ウェディング (wedding)	0.1398
	婚約 (engagement)	0.1391
	ドレス (dress)	0.1024

conference on Knowledge discovery and data mining, KDD '10, 2010.

[2] D. Agarwal and M. Gurevich. Fast top-k retrieval for model based recommendation. In *Proceedings of the fifth ACM international conference on Web search and data mining*, WSDM '12, 2012.

[3] M. Aly, A. Hatch, V. Josifovski, and V. K. Narayanan. Web-scale user modeling for targeting. In *Proceedings of the 21st international conference companion on World Wide Web*, WWW '12 Companion, 2012.

[4] A. Broder, M. Fontoura, V. Josifovski, and L. Riedel. A semantic approach to contextual advertising. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '07, 2007.

[5] A. Z. Broder, D. Carmel, M. Herscovici, A. Soffer, and J. Zien. Efficient query evaluation using a two-level retrieval process. In *Proceedings of the twelfth international conference on Information and knowledge management*, CIKM '03, 2003.

[6] D. Chakrabarti, D. Agarwal, and V. Josifovski. Contextual advertising by combining relevance with click feedback. In *Proceedings of the 17th international conference on World Wide Web*, WWW '08, 2008.

[7] H. Cheng and E. Cantú-Paz. Personalized click prediction in sponsored search. In *Proceedings of the third ACM international conference on Web search and data mining*, WSDM '10, 2010.

[8] H. Cheng, R. van Zwol, J. Azimi, E. Manavoglu, R. Zhang, Y. Zhou, and V. Navalpakkam. Multimedia features for click prediction of new ads in display advertising. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '12, 2012.

[9] K. S. Dave and V. Varma. Learning the click-through rate for rare/new ads from similar ads. In *Proceedings*

Table 6: Online A/B testing results. Metrics are click-through rate (CTR), cost per click (CPC), and revenue per request (RPR). Values represent the relative gains. We performed chi-squared test on the CTR results. * : p-value $<$ 0.05, ** : p-value $<$ 0.01, *** : p-value $<$ 0.001

Metric	Website							
	A	B	C	D	E	F	G	H
CTR	−3.67% **	+4.60% ***	+0.48%	+2.82% *	+2.47% **	+1.42%	+3.27%	+4.02% ***
CPC	+3.63%−	2.00%	+1.62%	+1.31%−	1.01%	+7.51%−	2.42%−2.94%	
RPR	−0.18%	+2.51%	+2.10%	+4.17%	+1.44%	+9.04%	+0.77%	+0.97%

of the 33rd international ACM SIGIR conference on Research and development in information retrieval, SIGIR '10, 2010.

[10] S. Ding and T. Suel. Faster top-k document retrieval using block-max indexes. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '11, 2011.

[11] M. Fontoura, V. Josifovski, J. Liu, S. Venkatesan, X. Zhu, and J. Zien. Evaluation strategies for top-k queries over memory-resident inverted indexes. In *Proceedings of the 37th International Conference on Very Large Data Bases*, volume 4, 2011.

[12] T. Graepel, J. Q. Candela, T. Borchert, and R. Herbrich. Web-scale bayesian click-through rate prediction for sponsored search advertising in microsoft's bing search engine. In *Proceedings of the 27th International Conference on Machine Learning*, 2010.

[13] A. Hatch, A. Bagherjeiran, and A. Ratnaparkhi. Clickable terms for contextual advertising. In *ADKDD*, 2010.

[14] T. Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '02, 2002.

[15] A. Joshi, A. Bagherjeiran, and A. Ratnaparkhi. User demographic and behavioral targeting for content match advertising. In *Proceedings of the fifth international workshop on Data mining and audience intelligence for advertising*, ADKDD '11, 2011.

[16] M. Karimzadehgan, W. Li, R. Zhang, and J. Mao. A stochastic learning-to-rank algorithm and its application to contextual advertising. In *Proceedings of the 20th international conference on World wide web*, WWW '11, 2011.

[17] K.-C. Lee, B. Orten, A. Dasdan, and W. Li. Estimating conversion rate in display advertising from past performance data. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '12, 2012.

[18] C. D. Manning, P. Raghavan, and H. Schtze. *Introduction to Information Retrieval.* Cambridge University Press, 2008.

[19] H. B. McMahan, G. Holt, D. Sculley, M. Young, D. Ebner, J. Grady, L. Nie, T. Phillips, E. Davydov, D. Golovin, S. Chikkerur, D. Liu, M. Wattenberg, A. M. Hrafnkelsson, T. Boulos, and J. Kubica. Ad click prediction: A view from the trenches. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '13, 2013.

[20] V. Murdock, M. Ciaramita, and V. Plachouras. A noisy-channel approach to contextual advertising. In *Proceedings of the 1st international workshop on Data mining and audience intelligence for advertising*, ADKDD '07, 2007.

[21] A. Ratnaparkhi. A hidden class page-ad probability model for contextual advertising. In *Workshop on Targeting and Ranking for Online Advertising at the 17th International World Wide Web Conference*, 2008.

[22] R. Rosales, H. Cheng, and E. Manavoglu. Post-click conversion modeling and analysis for non-guaranteed delivery display advertising. In *Proceedings of the fifth ACM international conference on Web search and data mining*, WSDM '12, 2012.

[23] Y. Tagami, T. Hotta, Y. Tanaka, S. Ono, K. Tsukamoto, and A. Tajima. Translation method of contextual information into textual space of advertisements. In *Proceedings of the Companion Publication of the 23rd International Conference on World Wide Web Companion*, WWW Companion '14, 2014.

[24] Y. Tagami, S. Ono, K. Yamamoto, K. Tsukamoto, and A. Tajima. Ctr prediction for contextual advertising: learning-to-rank approach. In *Proceedings of the Seventh International Workshop on Data Mining for Online Advertising*, ADKDD '13, 2013.

[25] I. Trofimov, A. Kornetova, and V. Topinskiy. Using boosted trees for click-through rate prediction for sponsored search. In *Proceedings of the Sixth International Workshop on Data Mining for Online Advertising and Internet Economy*, ADKDD '12, 2012.

[26] C. Wang, R. Raina, D. Fong, D. Zhou, J. Han, and G. Badros. Learning relevance from heterogeneous social network and its application in online targeting. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, SIGIR '11, 2011.

[27] W.-t. Yih and N. Jiang. Similarity models for ad relevance measures. In *MLOAD - NIPS 2010 Workshop on online advertising*, 2010.

[28] S. Yuan, A. Z. Abidin, M. Sloan, and J. Wang. Internet advertising: An interplay among advertisers, online publishers, ad exchanges and web users. *CoRR*, 2012.