TopicMachine: Conversion Prediction in Search Advertising Using Latent Topic Models

Ahmet Bulut, *Member, IEEE*

Abstract—Search Engine Marketing (SEM) agencies manage thousands of search keywords for their clients. The campaign management dashboards provided by advertisement brokers have interfaces to change search campaign attributes. Using these dashboards, advertisers create test variants for various bid choices, keyword ideas, and advertisement text options. Later on, they conduct controlled experiments for selecting the best performing variants. Given a large keyword portfolio and many variants to consider, campaign management can easily become a burden on even experienced advertisers. In order to target users in need of a particular service, advertisers have to determine the purchase intents or information needs of target users. Once the target intents are determined, advertisers can target those users with relevant search keywords. In order to formulate information needs and to scale campaign management with increasing number of keywords, we propose a framework called *TopicMachine*, where we learn the latent topics hidden in the available search terms reports. Our hypothesis is that these topics correspond to the set of information needs that best match-make a given client with users. In our experiments, TopicMachine outperformed its closest competitor by 41 percent on predicting total user subscriptions.

Index Terms—Internet advertising, topic modelling, search engine marketing

1 Introduction

The Internet has fostered on the ability to search the content people produce on the web and find those pages that are highly relevant to a given query. Lucrative markets are created out of the *information-seeking behavior* of billions of people traversing the web [1].

Consider a do-it-yourself (DIY) social network creation platform that sells subscription based services. There are two options for the company to market their services:

1) Launch a marketing campaign for displaying ads to the full set of Internet users, or
2) Employ what is known as precision targeting as in “show my ad to any user who enters the query ‘create my own social networking site’ ”.

The key insight in the second option is that a user query may indicate what is known as “purchase intent”. If a user types into a web search engine “create my own social networking site”, then we may infer that this user wants to create a social network, and is potentially willing to pay for the provided service. There could be many such users browsing the web. An advertiser can create an online advertisement that speaks exactly to the market segment containing these users or in other words create an advertisement to the specific purchase intent in question. This new type of advertising is attractive to both advertisers and customers at the same time. The query “create my own social networking site” is called a *search keyword*, or keyword for short, and the type of advertising that revolved around this notion is called keyword-based advertising. Pay per click (PPC) advertising and SEM are alternative names that are used to refer to the same type of advertising.

SEM agencies and practitioners manage thousands of search keywords on behalf of their clients. Any increase in the number of products being offered results in an increase in the number of search keywords being managed. Furthermore, as products tend to get more personal, catering to such niche segments increases the size of the keyword portfolio to manage dramatically. The campaign management dashboards provided by Google Adwords, Google Adwords Editor for bulk edits, Microsoft adCenter or Bing Ads have interfaces to change bid, scope, budget, and many other attributes per keyword or per groups of keywords. In addition, the management dashboards have features for advertisers to annotate their various bid choices, keyword ideas, and advertisement text options, and later on to conduct controlled experiments (A/B tests) on attribute variants [2]. The ability to run controlled experiments enables advertisers to test and select the best performing variants for their online marketing campaigns. However, given a large keyword portfolio and many variants to consider, the campaign management can easily become a burden on even experienced advertisers.

Before creating a search campaign, an advertiser first needs to identify information needs (purchase intents), expressible as search keywords [3]. These information needs are succinct descriptions of what each client product or service is offering to its users. For our example online platform to create white-label social networks, a mixture of information needs might look like “I want to create an online social community for fostering communication in our neighbourhood and for increasing awareness to environmental issues concerning the welfare of our neighbourhood. Furthermore, I want this online community to have private access”. This

• *The author is with the Department of Computer Science, İstanbul Şehir University, İstanbul 34662, Turkey, and Grou.ps Inc., Palo Alto 94403, CA. E-mail: ahmetbulut@sehir.edu.tr, ahmet.bulut@groups-inc.com.*

Manuscript received 17 July 2013; revised 18 Mar. 2014; accepted 18 Mar. 2014. Date of publication 25 Mar. 2014; date of current version 26 Sept. 2014.
Recommended for acceptance by P.G. Ipeirotis.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TKDE.2014.2313868

mixture of information needs might be transformed into a query such as "create a private social community for communication and collaboration". From an information retrieval perspective, a user is relevant if he needs a tool to create an online community, not because what he typed into a web search engine just happens to contain all the words in the query. This distinction is often misunderstood in practice, because the information need is not overt. But, nevertheless, an information need is present. The complexity of the problem comes from the fact that if a user types "python", they might want to know where they can purchase a pet python. Or they might want information on the Python programming language. It is very difficult to know what the hidden information need is from a few words in the query. The management dashboards fall short of providing a solution to formulate the hidden needs.

1.1 Motivation of the Paper

In order to match the users in need of a particular service with the client, which provides that service, advertisers have to determine the purchase intents or information needs of target users. Once the target intents are determined, advertisers can target those users with relevant search keywords. In order to compile a relevant set of search keywords, advertisers analyze search terms reports, search query logs, and trend reports provided by ad-brokers. In these reports, advertisers are exposed to how their target users express their hidden information need. Reviewing how users express their intent is very crucial as there is an impedance mismatch between how an advertiser describes an information need versus how a user expresses that need [4]. After reviewing reports, the advertiser may decide to add new search keywords to the portfolio. As a consequence, the size of the keyword portfolio keeps increasing over time. The portfolio may be pruned by deleting or pausing keywords that perform poorly [5], but the pruning has the risk of failing to capture the target information needs.

Given a large number of keywords, it is difficult to maintain keyword coherence within a campaign, and even more difficult to manage multiple campaigns consisting of a wide array of continuously evolving sets of keywords. Even though, management dashboards provide many features to slice and dice these keywords, they do not provide a semantic overlay or a topical structure on top of the existing campaigns. If such topical structures would exist, then an advertiser could easily manage these coherent topical structures rather than managing raw sets of keywords. This is critically important as more and more products and services get added to the product portfolio, the marketing of these offerings have to scale simultaneously. Hiring more people for the task is not a sustainable solution, and there are no guarantees for whether sales performance or return on investment (ROI) would also scale in line with such traditional marketing.

With a topical structure overlaid on top of the search campaigns, the advertisers can easily make campaign-wide changes, see the effects of such changes, understand the topical trends in the underlying search traffic over time, strategize into the future using the insights driven from structural trends, and target new market segments. All such management operations should be feasible while the underlying set of keywords scales to millions.

1.2 Contribution of the Paper

In order to scale SEM with an increasing number of product offerings while at the same time optimizing for conversions, we propose a framework called TopicMachine. In TopicMachine, we learn the latent topics hidden in the available search terms reports. Our hypothesis is that these topics correspond to the set of information needs that best matchmake the client with its users. For this purpose, we use a Latent Dirichlet Allocation (LDA) based topic model [6]. Since information needs may change over time or drift in concept, we learn dynamic topic models (DTM) [7] by sequentially chaining model parameters in a gaussian process across a well-defined epoch, e.g., weekly, bi-weekly, or monthly. In order to assess the quality of the models learned, we show the predictive power of the framework by measuring how well conversions per epoch can be predicted.

2 RELATED WORK

Probabilistic topic modeling is used to discover and annotate large archives of documents with thematic information [8]. With an increasing number of news, scientific articles, books, blogs, and web pages, it gets more difficult to categorize and search among the wealth of information. The idea behind LDA is to model documents as being generated from multiple topics (e.g., K topics) such that each of these topics is a probability distribution over a pre-defined vocabulary. Each document in the corpus exhibits topics in different proportions. In order to learn word distributions per topic and topic proportions per document, posterior probabilistic inference is used. With the observed documents in the corpus as output, hidden topical structure can be inferred by a deterministic variational method. In the variational inference procedure, a simpler distribution that contains free variational parameters is used to approximate the true posterior distribution. There are three sets of hidden variables each of which is governed by a different variational parameter: (1) a word distribution per topic, (2) a topic distribution per document, and (3) word-to-topic assignment per document [9]. All variables are assumed to be independent of each other. The variational parameters that maximize the log likelihood of the observations under the model are computed iteratively by continuous optimization using coordinate ascent.

The independence assumption in LDA has several drawbacks. For example, the presence of one latent topic may be correlated with the presence of another. Furthermore, since there is no notion of time in the model, documents in the corpus are exchangeable. However this assumption is inappropriate for many corpora including search terms reports. Search terms submitted by users reflect evolving content. In order to model time explicitly, dynamic topic models are introduced. In a DTM, each topic is dependent on its previous state and evolves in a state space model with Gaussian noise [7]. DTM has been successfully used to model thematic content change of scientific articles [10]. We use DTM over weekly search terms reports to learn an evolving conceptual model that can capture the hidden information needs and that can predict the sales performance better than alternative supervised approaches.

LDA effectively provides a mechanism to reduce dimension within a given text corpus. With a K-topic LDA model learned over the corpus, each document is expressed as a K-dimensional feature vector of probability values. The sum of these values add up to one. As such, these vectors can readily be used as features for classification, for similarity search as in similar topical proportions indicate similar documents, and for regression. Some textual data contain response variables such as movie reviews and their star ratings, comments and their total number of likes, and many other categorical responses. In such scenarios, the responses themselves can be modeled jointly with the associated documents in order to find the latent topics that best predict the response variable [11]. This model called supervised-LDA (sLDA) can be used for classification and regression tasks. The sLDA is used successfully to classify and annotate images [12]. In our settings, an sLDA model learned over weekly search terms reports and total weekly conversions for say the past 52 weeks can be used predict the future sales potential. This approach is implemented and considered as a competing technique to our proposed approach.

Another supervised extension of LDA is to model each document with only a subset of the topics instead of using the full set of topics. The subset of topics are essentially dictated by human-annotation, metadata, or other automatic tags that are deducible from domain specific characteristics such as the semi-structure present in the text. Labelled LDA (L-LDA) [13] has been shown to outperform Support Vector Machines (SVM) [14] on certain text summarization tasks, e.g., representative snippet extraction per tag. Generally, LDA models are used on whole document corpora. In order to demonstrate the efficacy of inferring topic models on short text corpora, L-LDA has been applied to characterize tweets [15]. On two information consumption tasks as (1) recommending users to follow and (2) identifying interesting tweets, L-LDA performance has been shown to improve when it is coupled with TF-IDF based information retrieval. This is because each of LDA and TF-IDF captures different aspects of textual similarity. Furthermore, L-LDA's classification and prediction performance is demonstrated on a large scale test conducted on 31.5M tweet data set [16]. L-LDA outperforms SVM on predicting top-k ($k \leq 3$) topics that represent a user's profile and on predicting the similarity of two user profiles. The superior performance of LDA on classifying and regressing shorter texts supports our approach's fidelity for the application of LDA on search keywords.

When dealing with a large number of topics, it is beneficial to enforce a network structure on the topics learned. Higher order relations between topics make the topic exploration easier as well as managing these topics and reasoning about them. Relational Topic Model (RTM) uses the distance between topic proportions for linking documents and for super-imposing a network structure on topics [17]. In order to generalize the integration of metadata (as in L-LDA) and other contextual information about documents such as author, time of writing, and place of publication into the topic models learned, kernel topic models (KTMs) can be used to further enhance the trustworthiness of the latent topics discovered [18].

The management of search keywords benefit from model driven visualizations for text analysis and text summarization. For example, finding out descriptive key phrases for text visualization can be used to create thematic structure over the search keywords. Statistical and linguistic features are combined within a logistic regression model to learn phrase ranks [19]. The top-N phrases are grouped using named entity recognition. In each group, a single phrase is selected to eliminate redundancy over the same entities. The final list of phrases are then displayed in a tag cloud. This work is complementary to our proposed approach as salient key phrases can be visualized in line with the topics discovered to communicate the textual content of the campaigns. In Termite [20], LDA is used to discover topics in conference papers. For each topic, salient terms that best distinguish the topic from the other remaining topics are identified. This approach is similar to popular TF-IDF scoring of terms, where TF component magnifies highly probable terms within topics, while IDF accentuates less common terms across topics. Sets of salient terms that are likely to co-occur are displayed close to each other in the term-topic matrix. This seriation step is done so that when the final term-topic matrix is displayed, the practitioner can inspect thematic clusters easily. As a cautionary note, supervised latent modeling that inherits domain characterization is observed to provide more interpretable and trustworthy visualizations compared to unsupervised latent topic discovery [21].

Concept hierarchies or topic clusters are used in search keyword recommendation. Given a set of advertisements and the set of keywords that they are subscribed to, a mixture of Bernoulli profiles are learned in a Bayesian framework to discover coherent clusters [22]. Later, cluster membership is used to assess whether a new and unknown keyword belongs to the general theme represented by that cluster. Even though the clusters correspond to topics in LDA, this work is different than LDA in such a way that mixed membership model with documents being generated from multiple topics is not used. Semi-supervised modeling approaches incorporate semantic relationships between keywords [23] instead of solely relying on unsupervised statistical information within the corpus. A keyword is first matched to a concept within Open Directory Project (ODP) ontology. Once the concept is found, the concept hierarchy within the ontology is used to find other concepts that are relevant to the primary concept. Each concept represents a set of webpages that are categorized under that concept within ODP. The webpage contents can then be used to suggest new keywords. To answer the question of how to best integrate a crowd-sourced ontology into unsupervised probabilistic modeling requires further research.

In similar vein to LDA, a generative translation model and a corresponding language model are used in SEM research to suggest search phrases for bidding [24]. The translation model formulates the probability of a given search term to align with a corresponding term in the target webpage. The higher is the alignment probability, the more relevant is the search term to the target webpage. The translation probabilities are learned iteratively using the

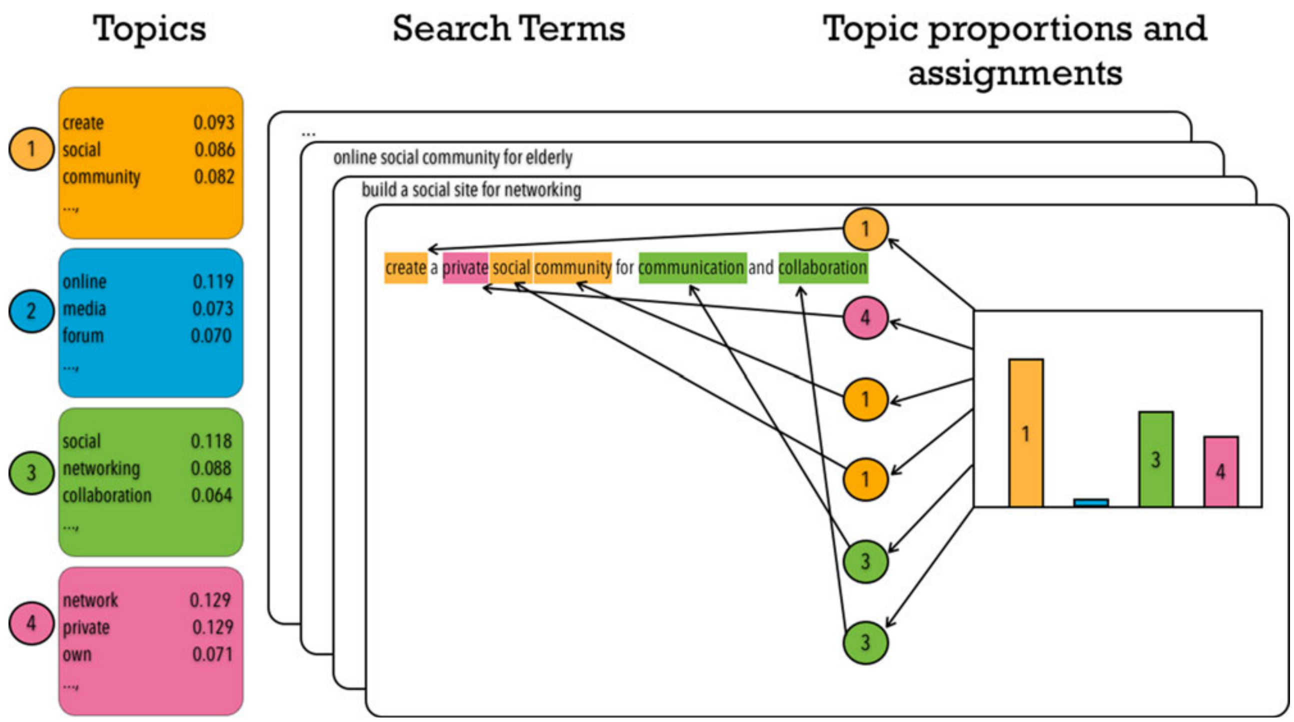


Fig. 1. The intuitions behind Latent Dirichlet Allocation. We assume that some number of “topics”, which are distributions over words, exist for the whole search terms collection (far left). Each search term is assumed to be generated as follows: First choose a distribution over the topics (the histogram at far right); then for each word, choose a topic assignment (the colored circles) and choose the word from the corresponding topic. The topics and topic assignments are illustrative.

Expectation Maximization (EM) algorithm. In order to preserve the translation probability mass that otherwise might be wasted due to impedance mismatch between page text and search terms, null tokens are added to be aligned with extraneous terms in the webpage. A bigram based language model is used to learn prospective query phrases while preserving term order. The combined models can be used as complementary to our work for generating non-intuitive search keywords for a given keyword portfolio.

LDA has been used to estimate the query advertisability in sponsored search. A predictive model is built using words found in queries, where the contribution of each word to advertisability is the fraction of times the word is present in queries that get user clicks versus queries that do not get user clicks [25]. LDA topical membership information is used as a binary feature for representing queries in a regression model. Contrary to our holistic approach of using LDA for representing all search terms of a given campaign per epoch, LDA is used for enriching feature extraction from individual queries.

Automatic management of advertising campaigns is studied extensively. For creating search campaigns targeting online product catalogs that contain thousands of products, automatic keyword generation from product specific landing pages help reduce the amount of setup work that advertisers otherwise need to do manually [26]. On a similar note, automatic generation of ad texts is studied for large-scale web advertising [27]. Sentence summarization and sentence scoring techniques are used to suggest descriptive ad texts. A template based approach is used to validate and conform the generated ad texts to the target ad platform. For campaign budget management, a genetic approximation algorithm to the optimal Knapsack solution is used to optimize a given campaign’s budget in real time using keyword level performance statistics [28]. In the presence of a large portfolio of keywords and multiple ad slots for each keyword, determination of optimal bids per keyword under a limited budget is done using constraint optimization with Lagrange multipliers. In the proposed model, the ratio of the marginal change in the advertiser’s expected revenue over the marginal change in the advertiser’s expected cost is taken to be constant across keywords [29].

3 Foundations and Methodology

In this section, we describe the basic ideas behind LDA [6] as it forms the building block in our framework. The intuition behind LDA is that search terms encompass multiple information needs. A search term also known as a search keyword is what a web user types in for querying a search engine. An advertisement keyword is what an advertiser uses for targeting search engine users. Keywords consist of tokens, each of which is a word of a natural language. For example, consider the search term “create a private social community for communication and collaboration” in Fig. 1. In this case, the user would like to create a social community, which should be private, and furthermore wants it to be used for enhancing community communication and collaboration. The words about creating online social communities, such as “create”, “social”, and “community” are highlighted in yellow and marked with number 1 to indicate topic 1; words about privacy in online social networks, such as “private”, are highlighted in pink and marked with number 4 to indicate topic 4; and words about social networking sites for collaboration, such as “communication” and "collaboration", are highlighted in green and marked with number 3 to indicate topic 3. With this insight, we can see that this search term blends private networks, social communities, and networking sites.

LDA assumes that there is a generative process that gave rise to the observed search terms. And the probabilistic inference is used to find out the hidden parameters that characterize the generation. A *topic* is a multinomial distribution over a chosen vocabulary of words found in the search terms. For example, *networking sites* topic has words about social networking with high probability, while *private networks* topic has words about privacy issues with high probability. The assumption here is that these topics have been specified before the data arose. With the topics specified, the algorithm for generating a search term is given in Algorithm 1. Since the algorithm itself is self explanatory, we omit the textual description. The goal of LDA is to find out the topic structure given the observed search terms. The topic structure consists of the topics themselves, per-document topic distributions, and per document per word topic assignments. The topic structure is the hidden structure. The hidden structure is more formally described as follows:

1) Let K denote the number of topics, and N denote the number of words in the dth search term.
2) Topics are $\beta_{1:K}$, where each β_k is a distribution over words in the vocabulary (see word distributions on the far left of Fig. 1).
3) The topic proportion distribution for the dth search term is θ_d, where $\theta_{d,k}$ is the frequency of topic k in search term d (see the histogram on the far right of Fig. 1).
4) The topic assignments for the words in the dth search term are z_d, where $z_{d,n}$ is the topic assignment for the nth word in the search term (see the coloured and numbered circles in Fig. 1).
5) The observed words for search term d are w_d, where $w_{d,n}$ denotes the nth word in d.

Algorithm 1 GenerateSearchTerm

```
Input: Topics (there are $K$ topics)
Output: topic assignments to words, topic proportions
1: N ← length of the search term.
2: topicProportions ← {topic$_1$:0, topic$_2$:0, ..., topic$_K$:0}.
3: topicAssignments ← {1: None, 2: None, ..., N: None}.
4: topicDist ← randomly choose a distribution over Topics.
5: for i in range(N) do
6:     topic ← sample a topic from topicDist.
7:     wordDist ← retrieve word distribution for topic.
8:     word ← randomly choose a word from wordDist.
9:     topicProportions[topic] ← topicProportions[topic] + 1
10:    topicAssignments[i] ← topic
11: end for
12: return topicProportions, topicAssignments.
```

With the mathematical notation outlined above, the LDA can be graphically represented and summarized as shown in Fig. 2. One can fit an LDA model on search terms D_t for a given time slice t. For the next time slice $t+1$, learning a new LDA model from scratch on search terms D_{t+1} discards the latent structure learned at time t. This approach inherently assumes that there is no temporal correlation between latent structures across time. However, this assumption is not correct for search campaigns due to a number of reasons:

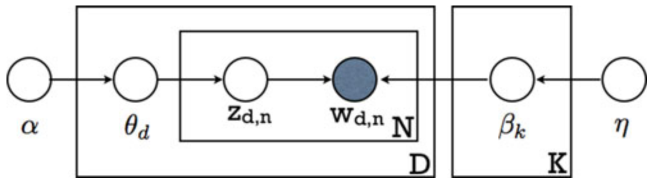


Fig. 2. The graphical model for LDA. Each node is a random variable. The hidden nodes for topic proportions, per term topic assignments, and topics themselves are shown as unshaded coins. The observed words for the search term are shown as shaded coins. The rectangles represent the replication in plate notation. For example, the N plate represents a search term, while K plate represents the topical distribution. The D plate represents the whole set of search terms. Hyper-parameters α and η represent pseudo-count smoothing on θ_d and β_k distributions respectively.

1) Current performance and quality of a search campaign affects its performance in the future. This is because ad-brokers continuously monitor campaign performance: its click through performance, the relevance of campaign keywords to client services provided, and the quality of these services. The quality in terms of user satisfaction is measured through implicit feedback, e.g., after-click round trip time, exit rate, and bounce rate.
2) The same users can return to site as return visitors.
3) Users can convert within a view through time frame of up to 30 days.

Therefore, discarding the notion of time in modeling does not capture the real time dynamics of search networks. Perhaps more importantly, the ability to incorporate time gives advertisers the power to make a campaign change and see the impact of that change in natural time order compared to adhoc and independent modeling. In order to incorporate the temporal correlation between latent structures, DTM is proposed where each topic evolves from its predecessor smoothly [9]. Fig. 3 shows the graphical model for DTM. The kth topic at time t is represented by $\beta_{k,t}$. The kth topic at time $t+1$ is represented by $\beta_{k,t+1}$, which evolves from $\beta_{k,t}$. For each time slice t in T, the search terms D_t are used for refining the hidden parameters at time t.

In the following, we first present our framework called TopicMachine, which is a DTM-based model to learn time varying topics on search terms observed within fixed epochs. In particular, we focus on how to build a TopicMachine for predicting campaign conversions (see Section 3.6) and for capturing trends between discovered topics and those conversions (see Section 3.7). After this brief introduction, we explain the details of each constructive step used in building the full model.

3.1 TopicMachine

Given search terms $D_1, D_2, \ldots, D_T$, a T-dimensional conversions vector $\mathbf{y}$, the chosen number of topics K, LDA hyperparameters α and η, and a selective method to use for

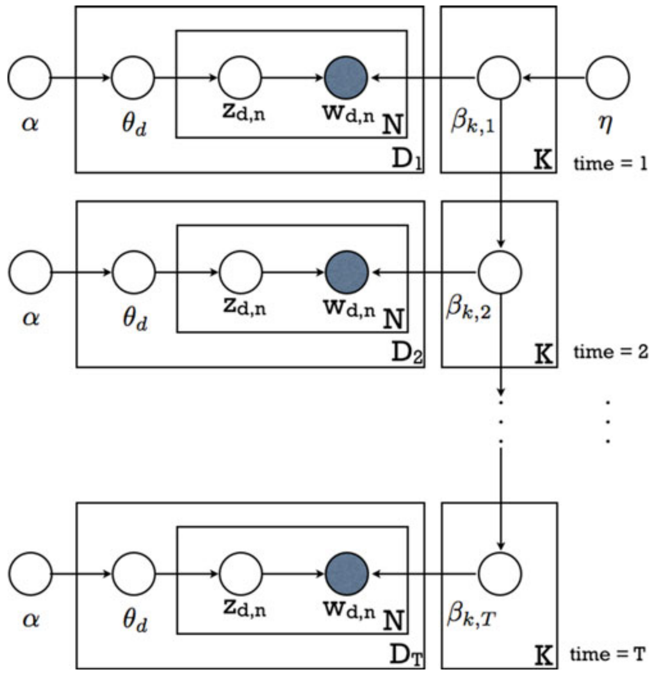


Fig. 3. The graphical model for DTM. Each topic evolves from its predecessor smoothly. The kth topic at time t is represented by $\beta_{k,t}$. The kth topic at time $t+1$ is represented by $\beta_{k,t+1}$, which evolves from $\beta_{k,t}$. The number of topics K is constant across time.

choosing the vocabulary V, a TopicMachine can be built as shown in Algorithm 2. First, the vocabulary V is determined on the union of all search terms at step 1. Inbetween steps 2-6, each search term collection D_t is formatted according to LDA-C format using V to construct input observations for the model. Using these observations at steps 7 and 8, a DTM model with K topics is learned for the input hyper parameter values of α and η. At step 9, features that represent topical density per epoch are extracted from the learned machine. The details of these features $\mathbf{X}$ are fully explained later in Section 3.5. Using $\mathbf{X}$ and conversion values input $\mathbf{y}$, a Lasso-based conversion predictor and correlation-coefficient based topical trends analyzer are learned at steps 10 and 11 respectively. The output at step 12 is the complete TopicMachine.

Algorithm 2 BuildTopicMachine

```
Input: Search terms $D_1, D_2, \ldots, D_T$, $\mathbf{y}$, $K$, $\alpha$, $\eta$, Selector
Output: TopicMachine
1: $V \leftarrow$ ChooseVocabulary($\bigcup_t D_t$, Selector)
2: Observations $\leftarrow$ []
3: for t in range($T$) do
4:     o $\leftarrow$ FormatLDA-C($D_t$, $V$)
5:     Observations $\leftarrow$ Observations + o
6: end for
7: $\beta$, $\theta$, $z \leftarrow$ LearnDTM(Observations, $K$, $\alpha$, $\eta$)
8: TopicMachine[Model] $\leftarrow \beta$, $\theta$, $z$
9: $\mathbf{X} \leftarrow$ ExtractFeatures(TopicMachine)
10: TopicMachine[Regressor] $\leftarrow$ TrainLasso($\mathbf{X}, \mathbf{y}$)
11: TopicMachine[Trends] $\leftarrow$ ComputeTrends($\mathbf{X}, \mathbf{y}$)
12: return TopicMachine
```

TABLE 1
Contingency Table for Word Occurrence and Conversion Events

	$e_{conversion} = 0$	$e_{conversion} = 1$
$e_{word} = 0$	N_{00}	N_{01}
$e_{word} = 1$	N_{10}	N_{11}

The event e_{word} denotes the occurrence of word. The event $e_{conversion}$ denotes the occurrence of a conversion. For example, the number of times a word occurred in a search term that converted is denoted by N_{11}. The number of times a word occurred in a search term without conversion is denoted by N_{10}.

3.2 Selection of Words for Vocabulary V

As each topic is a distribution over a fixed vocabulary V, the selection of words for the vocabulary affects the quality of the model fit. The vocabulary can be determined using various methods such as TF-IDF, χ^2 feature selection, or set-based collocation [3]. We describe each method next.

3.2.1 TF-IDF Based Selection

The tf-idf score for each word w in the full search terms collection D is computed as follows:

$$tf\text{-}idf_w = tf_w \times idf_w, \tag{1}$$

where idf_w equals to $\log\frac{|D|}{df_w}$. The value tf_w denotes the number of times w occurs in D, and df_w denotes the number of search terms that contain w. The words are sorted according to tf-idf scores in descending order. The $|V|$-many from top are selected as words in the vocabulary.

3.2.2 χ^2 Based Selection

Since the final model is used to predict the number of conversions across time slices, the words can be selected according to how well the occurrence of a word collides with the occurrence of conversion. If these two events are dependent, then the occurrence of the word makes the occurrence of the conversion more likely (or less likely), so it should be helpful as a feature. This is the rationale of χ^2 feature selection. Consider the contingency table in Table 1. The event e_{word} denotes the occurrence of word. The event $e_{conversion}$ denotes the occurrence of a conversion. For example, the number of times a word occurred in a search term that converted is denoted by N_{11}. The number of times a word occurred in a search term without conversion is denoted by N_{10}. By using the event counts, we can compute the χ^2 value per word w in D as $\chi^2(D, w)$

$$= \frac{(N_{11}+N_{10}+N_{01}+N_{00}) \times (N_{11}N_{00} - N_{10}N_{01})^2}{(N_{11}+N_{01}) \times (N_{11}+N_{10}) \times (N_{10}+N_{00}) \times (N_{01}+N_{00})}. \tag{2}$$

Words are sorted according to χ^2 values in descending order, and top-$|V|$ of them are selected into V.

3.2.3 Collocation Based Selection

Instead of using unigrams as words in V, one can use bigrams or trigrams to build V. Using higher order phrases may present opportunities to capture semantic relationships between words. In case of bigrams, if two words w_1 and w_2 tend to co-occur more than occurring individually, w_1w_2

pair represents a higher order building block to fit an LDA model on. Event count N_{11} reflect the number of bigrams in search terms that starts with w_1 followed by w_2. Event count N_{10} represents the number of occurrences of w_1 in search terms without being followed by w_2. Event count N_{01} represents the number of occurrences of w_2 in search terms without being preceded by w_1. With these values, the Jaccard association measure (intersection over union) of a given bigram w_1w_2 can be computed as

$$J(w_1, w_2) = \frac{N_{11}}{N_{11} + N_{10} + N_{01}}. \tag{3}$$

The bigrams (or trigrams) are sorted according to Jaccard scores in descending order. The top-$|V|$ are selected as words of the vocabulary.

3.3 Pre-Processing of the Data

Given a random permutation of the vocabulary, each word is first assigned to an index value equal to the word's rank in the permutation. Since there are $|V|$ words in V, word indices run from 1 to $|V|$. Each search term $d \in D_t$ is formatted according to what is known as the LDA-C format [6]. The given search term $d = w_1w_2 \dots w_N$ $(\forall\, w_n \in V)$ is represented as

$$N \quad i(w_1) : c(w_1) \quad i(w_2) : c(w_2) \quad \cdots \quad i(w_N) : c(w_N),$$

where N counts the total number of unique words of V in d, the function $i(w)$ returns the index of w in V, and the function $c(w)$ returns the number of occurrences of w in d. For $d =$"create a private social community for communication and collaboration", the words "a", "for", and "and" are not included in V as they are common stopwords. Then, $d' =$"create private social community communication collaboration" can be formatted as

$$6 \quad 3:1 \quad 51:1 \quad \cdots \quad 234:1,$$

where the index of word in V "create" is taken as 3, the index of word "private" is taken as 51, and the index of word "collaboration" is set to 234 for illustration.

3.4 Fitting a DTM Model

Each time slice $t = 1, \dots, T$ represents an epoch, e.g., one week or one month. Over the set of LDA-C formatted search terms $D_1, D_2, \dots, D_T$, a DTM model with hyper-parameters α and η is fit. The posterior probabilistic inference for LDA is to compute $p(\beta_{1:K}, \theta_{1:D_t}, z_{1:D_t} | w_{1:D_t})$, i.e., the conditional probability distribution of hidden structure given the observed search terms. Since the marginal probability of the search terms can be computed by summing over all possible hidden structures, this step is intractable and at best can be approximated. Variational approximation is used to approximate hidden variables as

$$\gamma_{d,n,k} \propto \frac{\#_{kw} - \gamma_{d,n,k} + \eta}{\#_k - \gamma_{d,n,k} + V\eta} \times (\#_{dk} - \gamma_{d,n,k} + \alpha),$$

where $\gamma_{d,n,k}$ is the assignment of topic k to the word position n in search term d. The $\gamma_{d,n,k}$ value on the right hand side refers to the $\gamma_{d,n,k}$ computed as the left hand side at previous iteration. By using the $\gamma_{d,n}$ distribution, we can compute the expected counts of how often each topic is paired with each word, i.e., $\#_{kw}$, and how often each search term is paired with each topic, i.e., $\#_{dk}$. More formally:

$$\#_{dk} = \sum_n \gamma_{d,n,k}, \tag{4}$$

$$\#_{kw} = \sum_{d,n} \gamma_{d,n,k} \times I(w_{d,n} = w), \tag{5}$$

$$\#_k = \sum_w \#_{kw}. \tag{6}$$

Since variational approximation is an iterative algorithm, it terminates after the change in model parameters between two successive iterations falls below a certain threshold. From the expected counts $\#_{kw}$ and $\#_{dk}$, we can easily approximate $\beta_{1:K}$ (topic distributions) and $\theta_{1:D_t}$ (topic proportions) respectively. Note that the expected counts $\#_{kw}$ computed at time slice t are used for learning at time slice $t+1$ in DTM.

3.5 Feature Vector Extraction

With all $\theta_{1:D_t}$ values computed for $t = 1, \dots, T$, we can represent each time slice t as a K-dimensional cumulative topic proportions vector $\mathbf{x_t} = \sum_{d \in D_t} \theta_d$. Since each search term is entered by a user after clicking on a campaign advertisement, $x_{t,k}$ represents the clicks weight of kth topic in D_t at time t for $k \in \{1, \dots, K\}$ and $t \in \{1, \dots, T\}$. The vector $\mathbf{X_k^T}$ represents a vector of T values, and corresponds to kth topic's clicks weight from time $t = 1$ to $t = T$. More formally,

$$\mathbf{X} = \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,K} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,K} \\ \vdots & \vdots & \ddots & \vdots \\ x_{T,1} & x_{T,2} & \cdots & x_{T,K} \end{bmatrix} \quad \text{and} \quad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_T \end{bmatrix},$$

where $\mathbf{y}$ is the conversions vector. The value y_t represents the total number of conversions at time slice t.

3.6 Fitting a Regression Model

In order to predict conversions that occur in a given time slice, we can train a linear model on $\mathbf{X}$ and $\mathbf{y}$ with L_1 prior as regularizer, aka Lasso (Least Absolute Shrinkage and Selection Operator) [30]. The optimization objective for Lasso is

$$\frac{1}{2 \times T} \times \| \mathbf{y} - \mathbf{X}\mathbf{r} \|_2^2 + \alpha \times \| \mathbf{r} \|_1, \tag{7}$$

where K-dimensional regression coefficients vector $\mathbf{r} = [r_1, \dots, r_K]$ represent weights fit for each topic. For a given cumulative topic proportions vector $\mathbf{x}$ representing a specific campaign budget allocation or a future marketing strategy, the Lasso model can be used to predict the total number of conversions.

3.7 Finding Similar Trends

The euclidean distance in z-norm space between two feature vectors is proportional to the strength of their correlation, or their trend similarity. First, each column of $\mathbf{X}$ is z-normalized to capture correlations between that topic and the number of conversions. Given $\mathbf{X_k^T} = [x_{k,1}, x_{k,2}, \ldots, x_{k,T}]$, $\mathbf{\hat{X}_k^T}$ can be computed as

$$\mathbf{\hat{X}_k^T}[t] = \frac{\mathbf{X_k^T}[t] - \mu(\mathbf{X_k^T})}{\sqrt{\sum_{t=1}^{t=T} \mathbf{X_k^T}[t]^2 - \mu(\mathbf{X_k^T})^2}}. \tag{8}$$

Similarly, $\mathbf{y}$ can be z-normalized by

$$\hat{\mathbf{y}} = \frac{y_t - \mu(\mathbf{y})}{\sqrt{\sum_{t=1}^{t=T} y_t^2 - \mu(\mathbf{y})^2}}. \tag{9}$$

Given $\mathbf{\hat{X}_k^T}$ and $\hat{\mathbf{y}}$, the correlation coefficient between them is equal to:

$$corr(\mathbf{\hat{X}_k^T}, \hat{\mathbf{y}}) = 1 - \frac{\| \mathbf{\hat{X}_k^T}, \hat{\mathbf{y}} \|_2^2}{2}. \tag{10}$$

Values of $corr > 0.95$ indicate strong positive correlations; while $corr < 0$ indicates negative correlation. A value of 0 indicates no trend correlation. Trends between topics and conversions across time present ample opportunity for advertisers to visualise and track real time campaign performance with respect to superimposed topical structure as demonstrated in Section 4.5.

3.8 Competing Techniques

In this section, we present three supervised competing techniques for predicting conversions. The first method is the adaptation of sLDA [11] to learn search campaign conversion performance of search terms. In the second method, search terms are grouped under enclosing marketing campaigns determined by the advertiser during campaign creation. In the third and the final method, the relevance feedback provided by the advertiser is used to build a conversion model.

3.8.1 Supervised-LDA (sLDA)

The response variable denoting conversions $\mathbf{y}$ is modelled jointly with the associated search terms $D_1, \ldots, D_T$ in order to find the latent topics that best predict the response variable [11]. For each D_t where $t \in \{1, \ldots, T\}$, we aggregate the search terms into a single LDA-C formatted row as

$$C_t \quad i(w_1) : c(w_1) \quad i(w_2) : c(w_2) \quad \cdots \quad i(w_{C_t}) : c(w_{C_t}),$$

where each word w_i is in $\bigcup_{d \in D_t} \bigcup_{w_{d,n} \in V} w_{d,n}$, and C_t denotes the cardinality of this set of unique terms in V that occur across all search terms within time slice t. In order to predict conversions, we train a supervised sLDA on the LDA-C formatted T rows and $\mathbf{y}$.

3.8.2 Per Campaign Clicks (PCC)

Since advertisers group together "coherent" search keywords within well defined campaigns,[1] we can aggregate total clicks received per campaign per epoch $t \in \{1, \ldots, T\}$. This supervised approach is in contrast to our LDA based topical structure or grouping of search terms, where groups are built without any supervision except the usage of observed search terms. If we denote the number of search campaigns by K, then this method also produces a $K \times T$ matrix $\mathbf{X}$ of feature vectors. A Lasso model to predict conversions is fit on $\mathbf{X}$ and $\mathbf{y}$ as explained in Section 3.6.

1. An online search campaign may contain multiple subgroups of keywords, which are called *adgroups*.

3.8.3 Retrieval Status Value (RSV)

Advertisers periodically review search terms reports provided by ad-brokers. The review process is done in order to label search terms as either relevant or nonrelevant for the search campaign target. Given the full set of labeled search terms D, we can compute a retrieval status value for a given search term $d \in D$ as follows:

$$RSV_d = \sum_{w \in d} \log \frac{p_w(1 - u_w)}{u_w(1 - p_w)}, \tag{11}$$

where p_w represents the probability of w appearing in relevant search terms, while u_w represents the probability of w appearing in nonrelevant search terms [3]. If we aggregate the RSVs of search terms per epoch $t \in \{1, \ldots, T\}$, then we get a $1 \times T$ matrix $\mathbf{X}$ of feature vectors. Similar to the previous method, a Lasso model is fit on $\mathbf{X}$ and $\mathbf{y}$ in order to predict conversions as explained in Section 3.6. RSV-based methods were used successfully in user profile generation where the interests of consumers are represented with a select set of keywords from user feedbacks [31].

4 Experimental Evaluation

In this section, we present the results of our empirical evaluation of TopicMachine, sLDA, PCC, and RSV on a real campaign data set. After briefly describing the data set, we introduce the test methodology and metrics used for comparing techniques. Then, comparative results are presented followed by a comprehensive study on TopicMachine's sensitivity to various parameters from the selection of vocabulary to the number of topics.

All development is done using Python and publicly available machine learning libraries NLTK [32] and Scikit-learn [33]. The DTM and sLDA code written in C++ is publicly available at http://www.cs.princeton.edu/blei/topicmodeling.html.

4.1 Campaign Data Set

The search terms data used in the experiments corresponds to data collected between June 20, 2012 and June 12, 2013. There were four different campaigns with $13{,}898$ unique search terms that resulted in 432 conversions in total. Within this time frame, $29{,}821$ clicks were received out of a total of $2{,}382{,}317$ ad impressions. In the experiments, the duration of a *week* is chosen as the epoch, which resulted in $T = 50$ different search term collections. Starting from 2012-06-18 and ending at 2013-06-10, each D_t for $t \in \{1, \ldots, 50\}$ corresponds to a week in strict time order.

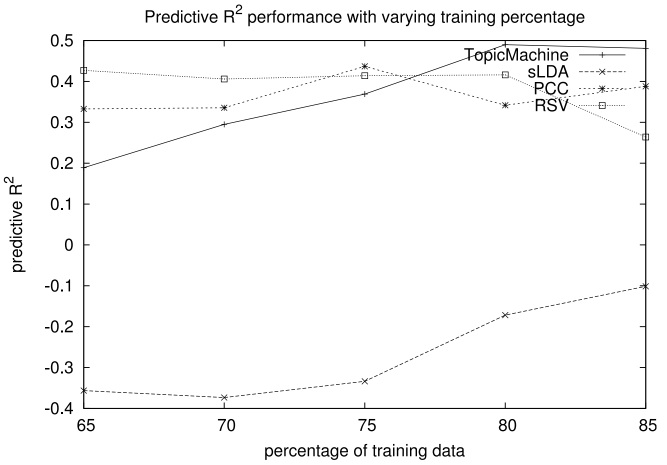


Fig. 4. Predictive R^2 performance with respect to varying the size of the training set. LDA model parameters K, $|V|$, α, and η are set to 5, 250, 0.01, and 0.01 respectively.

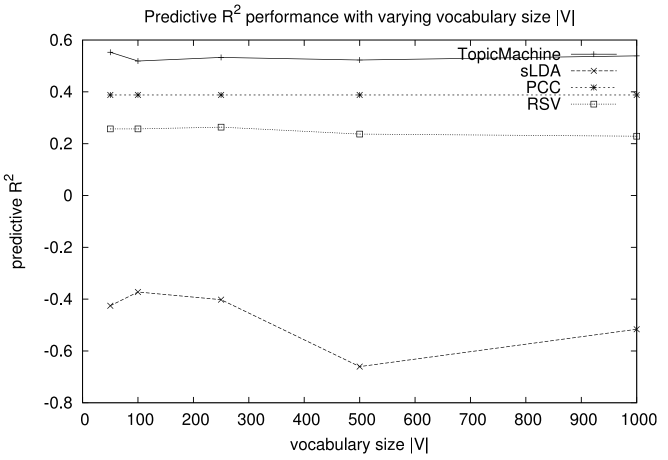


Fig. 5. Predictive R^2 performance with respect to varying vocabulary size. LDA model parameters K, α, and η are set to 10, 0.01, and 0.01 respectively. The test set corresponds to the 15 percent of the data.

4.2 Evaluation Philosophy

Since DTM mandates a strict time ordering, cross validation is performed for a hold-out set that corresponds to a certain tail percentage of the experiment period. For example, if 80 percent of the data is used for training and the remaining 20 percent is used for testing, then the test set corresponds to $D_{41}, D_{42}, \ldots, D_{50}$ since $50 \times 20/100 = 10$. The value 10 corresponds to the test offset. For topic models TopicMachine and sLDA, the results are computed by averaging the performance of 20 independent runs. The performance metrics that we measure are Mean Squared Error regression loss (MSE) and Predictive R^2, aka the coefficient of determination. The full set of feature vectors $\mathbf{X}$ is first divided into two subsets as a training set and a test set. Then, a Lasso based predictor is learned on the training set as explained in Section 3.6. Using the predictor, a conversion prediction is made for each data point in the test set. The predicted values are compared with their actual values to compute the performance metrics. More formally, given that the predicted values are denoted by $\mathbf{f}$ and the actual values are denoted by $\mathbf{y}$, then MSE and R^2 can be computed as

$$MSE = \sqrt{\sum_{i=T-offset+1}^{T} (y_i - f_i)^2}, \tag{12}$$

$$R^2 \equiv 1 - \frac{\sum_{i=T-offset+1}^{T} (y_i - f_i)^2}{\sum_{i=T-offset+1}^{T} (y_i - \mu(\mathbf{y}))^2}, \tag{13}$$

where *offset* corresponds to the test offset.

4.3 Comparative Results

In this section, we present the R^2 performance of TopicMachine, sLDA, PCC, and RSV with varying training data percentage, vocabulary size, and number of topics fit. LDA-based model hyper-parameters α and η are fixed at 0.01. The vocabulary selector is set to TF-IDF by default unless otherwise stated. With this setting, if the vocabulary size is set to $|V| = 500$, then the vocabulary contains 500 unigrams with the highest TF-IDF scores. By default, the test set corresponds to the last 15 percent of the available $T = 50$ data points. Therefore, the regression performance is measured on the last 15 percent of the data.

4.3.1 R^2 with Varying Percentage of Training Data

In order to test the dependency of the models learned on the available training data, we experiment with increasing training data percentage. Fig. 4 shows the results for training percentages from 65 to 85 percent. The performances of sLDA and TopicMachine improved dramatically when there was more training data available. These results indicate that topic models require a larger portion of the data set for training to achieve an acceptable accuracy. The performance of PCC initially fluctuated, but settled at a higher predictive power for 85 percent than for 65 percent. The RSV method performed better compared to all other techniques when less training data was available.

4.3.2 R^2 with Varying Vocabulary Size $|V|$

Since the choice of vocabulary affects the topics learned, we experiment with vocabulary size $|V|$ and how it affects the predictive R^2 performance. The number of topics is set to $K = 10$. The performance of PCC does not depend on $|V|$, and therefore is constant. The values for the vocabulary size varied from 50 to 1,000. The results are shown in Fig. 5. Even though PCC and RSV are supervised approaches based on domain expertise, they remain coarser at modeling intrinsic information need characteristics in comparison to TopicMachine. Superior performance of TopicMachine over sLDA at time series prediction indicates that modeling time explicitly pays off. For $|V| = 50$, TopicMachine achieved an R^2 of 0.55 and is nearly 41 percent better than its closest competitor PCC, which had an R^2 of 0.39. All methods performed at their best for $|V| = 50$, which indicates that the top fifty unigrams maintain most of the predictive power.

4.3.3 R^2 with Varying Number of Topics K

In order to quantify the effect of the number of topics K on the quality of the LDA-based models, we experiment with varying K and measure the predictive R^2 performance. The values for K varied from 5 to 25. The vocabulary size is set

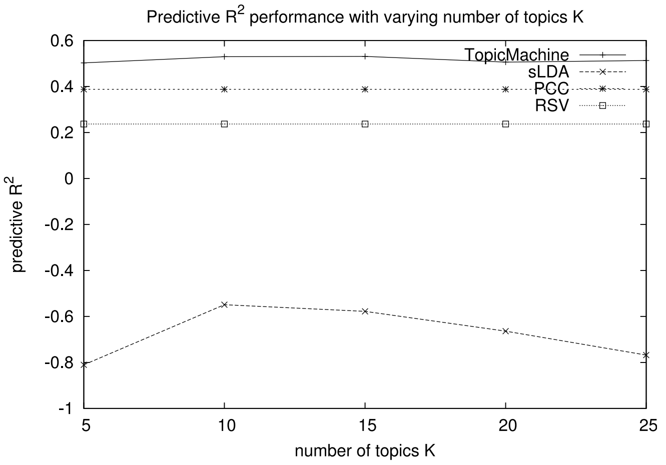


Fig. 6. Predictive R^2 performance with respect to varying number of topics K. LDA model parameters V, α, and η are set to $500, 0.01$, and 0.01. The test set corresponds to the 15 percent of the data.

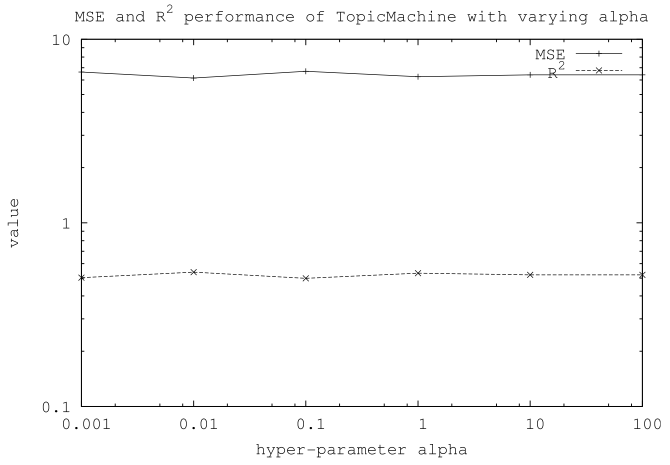


Fig. 7. MSE and Predictive R^2 performance of TopicMachine with respect to varying hyper-parameter α. The number of topics and vocabulary size are set to $K = 20$ and $|V| = 500$ respectively. LDA model parameter η is set to 0.01. The test set corresponds to the 15 percent of the data.

to $|V| = 500$. The performances of PCC and RSV do not depend on K, and therefore are constant. The results are shown in Fig. 6. As the number of topics increase, LDA-based models become more adept at capturing the hidden internal structure with TopicMachine always outperforming sLDA, PCC, and RSV. Even though sLDA achieves a performance of $R^2 = 0.63$ for $K = 20$ and $|V| = 500$ at $\alpha = 0.01$ in a particular run, its average performance is not as robust as TopicMachine. The average R^2 and the variance (μ, σ^2) for sLDA were measured as $(-1.14, 0.74)$, $(-0.73, 0.74)$, $(-1.20, 0.91)$, $(-0.68, 0.54)$, $(-0.94, 0, 59)$ for $K = 5, 10, 15, 20,$ and 25 respectively. With the variance in performance considered, the accuracy of sLDA stayed suboptimal compared to TopicMachine.

4.4 TopicMachine's Sensitivity to Parameters

In this section, the effect of vocabulary selectors and hyper-parameter α on TopicMachine's regression performance is quantified.

4.4.1 MSE and R^2 with Varying Vocabulary Selectors

As each topic is a distribution over a fixed vocabulary V, the selection of words for the vocabulary may affect the quality of the model fit. For this purpose, we experiment with three different vocabulary selectors as TF-IDF, Collocation (Jaccard score), and χ^2 selection. Before the experiment, the number of topics and vocabulary size are set to $K = 20$ and $|V| = 500$ respectively. The experimental results are shown in Table 2. Using 500 bigrams with the highest Jaccard scores slightly improves the prediction performance over 500 unigrams with the highest TF-IDF scores. This result shows the potential value one might gain using higher-order phrases in the vocabulary. Due to the size of the data set, we could not obtain more than 500 "meaningful" trigrams; therefore, studying the effect of trigrams on the model quality requires further evaluation on larger data sets. The same also holds for χ^2 selector due to the limited number of words within 95 percent confidence interval.

TABLE 2
MSE and Predictive R^2 Performance of TopicMachine with Respect to Varying Vocabulary Selectors

	MSE	$Predictive\ R^2$
TF-IDF unigrams	6.398	0.521
Jaccard bigrams	6.367	0.523
Jaccard trigrams	8.746	0.345
χ^2 bigrams	7.95	0.405

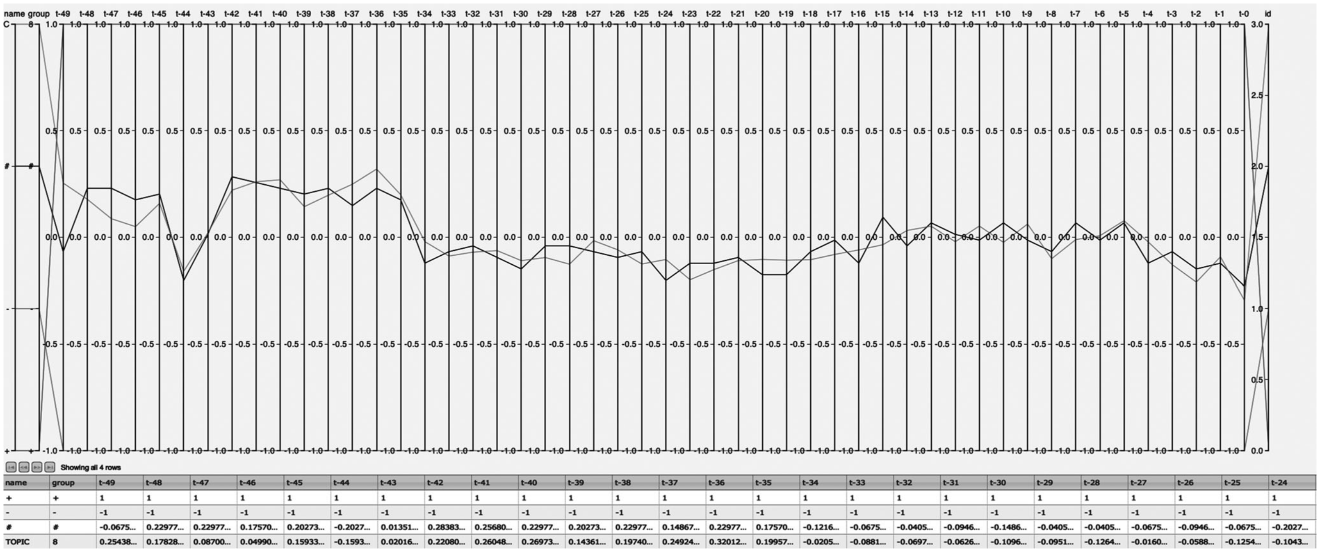
The alternative selectors are TF-IDF, Collocation (Jaccard score), and χ^2 selection. The number of topics and vocabulary size are set to $K = 20$ and $|V| = 500$ respectively. LDA model parameters α and η are set to 0.01. The test set corresponds to the 15 percent of the data.

4.4.2 MSE and R^2 with Varying α

The hyper-parameter α determines the initial starting points for topic proportions, and therefore it is mandatory to study whether changing α changes the quality of the model fit. For this experiment, the number of topics and vocabulary size are set to $K = 20$ and $|V| = 500$ respectively. The value of α is varied from 0.001 to 100. The results are shown in Fig. 7. The quality of prediction is similar for different values of α.

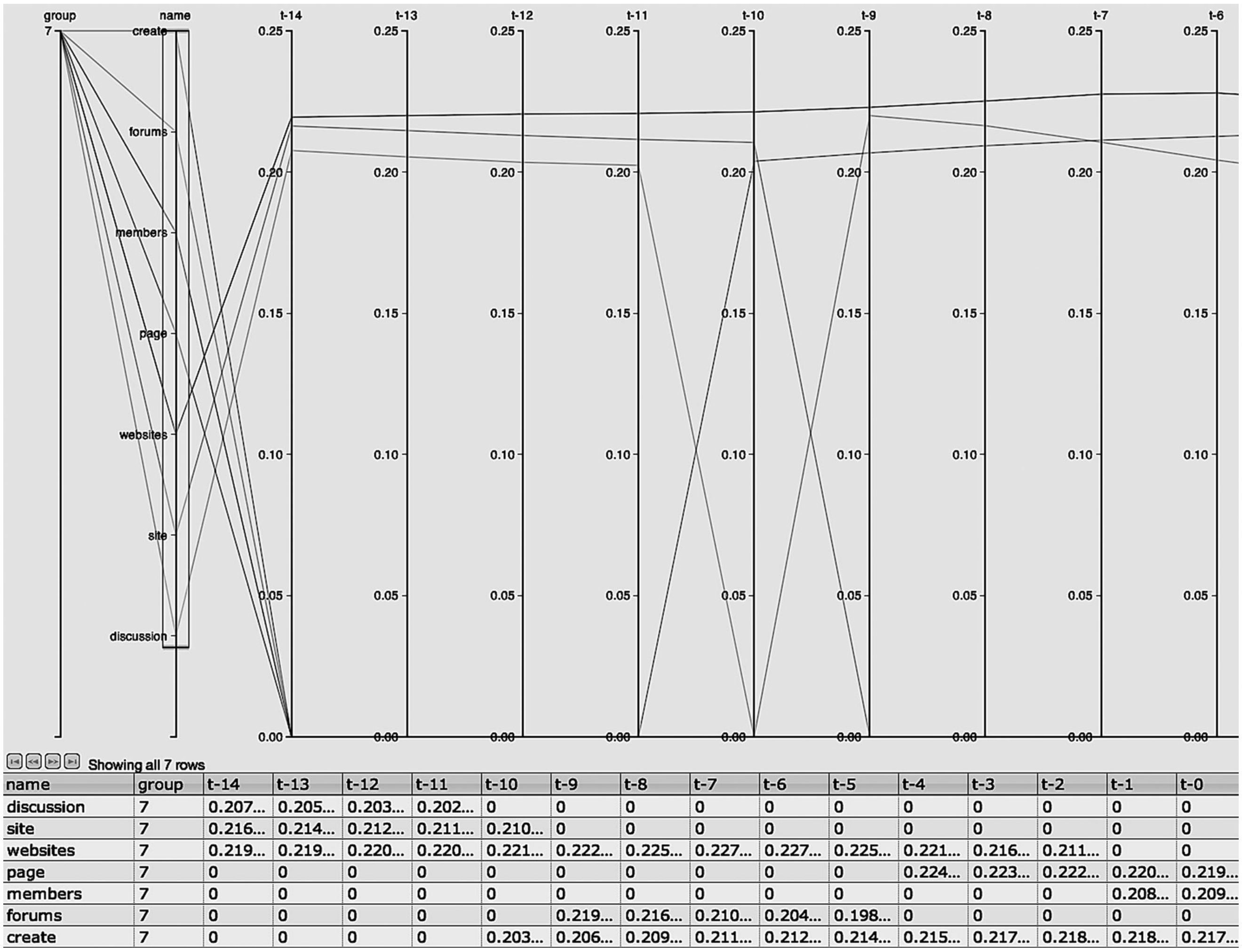
4.5 Riding Trends with TopicMachine

Using the z-normalised values of $\mathbf{X}$ and $\mathbf{y}$, TopicMachine provides an interface to capture similar trends inbetween topics, and between topics and total conversions. Fig. 8 shows the most correlated topic to conversions. The 8th topic within $K = 10$ topics on the real campaign data set was the best predictor of conversions. In this snapshot, only the eighth topic and total conversions (as #) were depicted. Fig. 9 shows a snapshot of how the seventh topic's most prominent word probabilities change across nine time slices or epochs in Parallel Coordinates [34]. The advertiser can visualise the full set of topics in the same pane or can use the brush feature to create a sliding window for filtering out data outside the range of interest. Furthermore, multiple brushes can be created at once. The visualisation interface of TopicMachine is built on top of parallel coordinates within Data Driven Documents (D3) [35]. Each coordinate corresponds to one of the T epochs.



name	group	t-49	t-48	t-47	t-46	t-45	t-44	t-43	t-42	t-41	t-40	t-39	t-38	t-37	t-36	t-35	t-34	t-33	t-32	t-31	t-30	t-29	t-28	t-27	t-26	t-25	t-24
+	+	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
-	-	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
#	#	-0.0675...	0.22977...	0.22977...	0.17570...	0.20273...	-0.2027...	0.01351...	0.28383...	0.25680...	0.22977...	0.20273...	0.22977...	0.14867...	0.22977...	0.17570...	-0.1216...	-0.0675...	-0.0405...	-0.0946...	-0.1486...	-0.0405...	-0.0405...	-0.0675...	-0.0946...	-0.0675...	-0.2027...
TOPIC	8	0.25438...	0.17828...	0.08700...	0.04990...	0.15933...	-0.1593...	0.02016...	0.22080...	0.26048...	0.26973...	0.14361...	0.19740...	0.24924...	0.32012...	0.19957...	-0.0205...	-0.0881...	-0.0697...	-0.0626...	-0.1096...	-0.0951...	-0.1264...	-0.0160...	-0.0588...	-0.1254...	-0.1043...

Fig. 8. Capturing trends with TopicMachine. The most correlated topic to conversions is the eighth topic in an example model with $K = 10$ topics on the real campaign data set used in experiments. Each parallel coordinate corresponds to one of the T epochs. In this snapshot, only the eighth topic and total conversions (as #) are depicted.



name	group	t-14	t-13	t-12	t-11	t-10	t-9	t-8	t-7	t-6	t-5	t-4	t-3	t-2	t-1	t-0
discussion	7	0.207...	0.205...	0.203...	0.202...	0	0	0	0	0	0	0	0	0	0	0
site	7	0.216...	0.214...	0.212...	0.211...	0.210...	0	0	0	0	0	0	0	0	0	0
websites	7	0.219...	0.219...	0.220...	0.220...	0.221...	0.222...	0.225...	0.227...	0.227...	0.225...	0.221...	0.216...	0.211...	0	0
page	7	0	0	0	0	0	0	0	0	0	0	0.224...	0.223...	0.222...	0.220...	0.219...
members	7	0	0	0	0	0	0	0	0	0	0	0	0	0	0.208...	0.209...
forums	7	0	0	0	0	0	0.219...	0.216...	0.210...	0.204...	0.198...	0	0	0	0	0
create	7	0	0	0	0	0.203...	0.206...	0.209...	0.211...	0.212...	0.214...	0.215...	0.217...	0.218...	0.218...	0.217...

Fig. 9. A snapshot of how the seventh topic's most prominent word probabilities change across nine time slices or epochs, which are represented by nine parallel coordinates.

5 Discussion and Future Work

In this section, we discuss the learnings we obtained from the experimental comparison of TopicMachine with the competing approaches sLDA, PCC, and RSV. Even though sLDA alone can achieve a high predictive accuracy in individual runs, its performance is not consistent across independent runs. In order to exploit the best properties of each LDA-based model (DTM-based TopicMachine and sLDA with response supervision), observations, time, and response variable can be modelled jointly in similar vein to Kernel Topic Models [18]. KTMs provide a generic framework where annotations or expert opinion from different information sources can be leveraged to learn probabilistic models that are driven with expert knowledge. It would be interesting to study how a DTM & sLDA model combination performs in comparison to a KTM-based model in SEM campaign management tasks such as conversion optimisation, keyword management, and precision targeting.

The performance of LDA-based models depends on the choice of the initial starting point. Some model fits may get stuck in local optima. Therefore, DTM-based approaches pose a great advantage in terms of evolving from an already established and plausible model fit. First, an sLDA model with a superior predictive power is found after repeated iterations. After that, a DTM-based model may evolve from it thereafter.

SEM Advertisers repeatedly review search terms reports to identify relevant or non-relevant search terms. Relevant search terms are used to expand existing keyword campaigns while non-relevant search terms are used to reduce the size of these campaigns. This problem is a classification problem on the search term space. An adaptation of sLDA with supervision in terms of relevance can be used to build a topic-based classifier for asserting relevance judgements automatically. Furthermore, Labeled-LDA [13] can be used with external labels such as hierarchical concepts in an ontology (ODP, Wikipedia, and DBpedia) to refine and guide statistical learning of topical structures on top of search terms. One way to incorporate supervision is as follows:

$$\gamma_{d,n,k} \propto \frac{\#_{kw} - \gamma_{d,n,k} + \eta}{\#_{k} - \gamma_{d,n,k} + V\eta} \times (\#_{dk} - \gamma_{d,n,k} + \alpha) \times I[k \in \Lambda_d], \tag{14}$$

where Λ_d stands as the supervision in terms of a known or specific set of labels for dth search term. With this restriction, dth search term is assumed to contain or be generated from the subset of topics in Λ_d instead of the whole set of K topics.

We considered trends between topics and trends between a topic and total conversions. There are many opportunities present in effective visualisation of TopicMachine models with external knowledge bases and ontologies. A knowledge base provides semantic relationships between entities, while statistical models offer what is latent in particular observations of those entities in various domains of interest. It would be interesting to explore how to visualise SEM campaigns together with hidden information needs, semantics, sales performance, and user experience for providing a comprehensive campaign management dashboard.

The superimposition of hierarchies on topical structures discovered by a TopicMachine, e.g., RTMs [17], presents an organisational opportunity to govern an online marketing team. For example, hierarchies naturally lend themselves to manage marketing activities at different levels of expertise or rank. While a marketing manager decides at a broader level how much marketing budget each information need is allocated to at a higher topic depth, a lower-ranked marketer can decide on how to divide her allocated budget within different subtopics at a lower depth downstream. This approach leads to denser hierarchies with more marketers or looser hierarchies with less marketers.

6 Conclusion

We proposed a framework called TopicMachine where we learn the latent topics hidden in the available SEM search terms data. Our foundational hypothesis is that there are a set of information needs hidden in and behind the search terms as a collection; the latent topics that are discovered through probabilistic inference over this collection correspond to these information needs. The topical structure stands as a viable tool to manage SEM campaigns with precise targeting of users in terms of relevance and to optimise for conversions.

TopicMachine uses an LDA-based topic model. Since information needs may change over time or drift in concept, we learn dynamic topic models by sequentially chaining model parameters in a gaussian process across a well-defined epoch. We assessed the quality of the models learned in TopicMachine by showing the predictive power of the framework. We measured how well the conversions can be predicted with features extracted from TopicMachine model. In experimental results, TopicMachine outperformed its closest competitor with an improvement of 41 percent in predictive R^2 performance.

Since TopicMachine's internal model can be used to reduce dimensions of the search term space, we foresee that advertisers can scale their campaign management to thousands of keywords comfortably with the use of TopicMachine while at the same time optimising for conversions. As a campaign manager, TopicMachine can be used to visualise topical structure at a given epoch, visualise how topical structure changes across time, predict conversion performance, capture trends between topics and conversions, generate search terms that conform to a particular topical distribution, allocate budget per topic, and other use cases that advertisers may see fit.

Acknowledgments

The author would like to thank Grou.ps Inc. for providing the campaign data set, Sean Gerrish from Sift Science for helping out with DTM compilation, Kai Chang for the implementation of parallel coordinates on D3, and Chong Wang from Carnegie Mellon University and David M. Blei from Princeton University for making the LDA implementations publicly available.

REFERENCES

[1] D. Easley and J. Kleinberg, *Networks, Crowds, and Markets: Reasoning about a Highly Connected World*. New York, NY, USA: Cambridge Univ. Press, 2010.

[2] R. Kohavi, R. Longbotham, D. Sommerfield, and R. M. Henne, "Controlled experiments on the web: Survey and practical guide," *Data Mining Knowl. Discovery*, vol. 18, no. 1, pp. 140–181, 2009.

[3] C. D. Manning, P. Raghavan, and H. Schutze, *Introduction to Information Retrieval*. New York, NY, USA: Cambridge Univ. Press, 2008.

[4] B. Ribeiro-Neto, M. Cristo, P. B. Golgher, and E. S. de Moura, "Impedance coupling in content-targeted advertising," in *Proc. 28th ACM Int. Conf. Res. Develop. Inform. Retrieval*, Aug. 2005, pp. 496–503.

[5] P. Rusmevichientong and D. P. Williamson, "An adaptive algorithm for selecting profitable keywords for search-based advertising services," in *Proc. 7th ACM Conf. Electron. Commerce*, Jun. 2006, pp. 260–269.

[6] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *J. Mach. Learning Res.*, vol. 3, pp. 993–1022, Mar. 2003.

[7] D. M. Blei and J. D. Lafferty, "Dynamic topic models," in *Proc. 23rd ACM Int. Conf. Mach. Learning*, 2006, pp. 113–120.

[8] D. M. Blei, "Probabilistic topic models," *Commun. ACM*, vol. 55, no. 4, pp. 77–84, 2012.

[9] D. M.w Blei and J. D. Lafferty, "Topic models," in *Text Mining: Classification, Clustering, and Applications*, London, U.K.: Chapman & Hall/CRC Press, 2009 pp. 71–94.

[10] S. Gerrish and D. M. Blei, "A language-based approach to measuring scholarly impact," in *Proc. 27th Int. Conf. Mach. Learning*, 2010, pp. 375–382.

[11] D. M. Blei and J. McAuliffe, "Supervised topic models," in *Proc. Advances Neural Inform. Process. Syst. 20*, 2008, pp. 121–128.

[12] C. Wang, D. Blei, and F.-F. Li, "Simultaneous image classification and annotation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2009, pp. 1903–1910.

[13] D. Ramage, D. Hall, R. Nallapati, and C. D. Manning, "Labeled LDA: A supervised topic model for credit attribution in multi-labeled corpora," in *Proc. Conf. Empirical Methods Natural Language Process.*, 2009, pp. 248–256.

[14] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learning*, vol. 20, no. 3, pp. 273–297, Sep. 1995.

[15] D. Ramage, S. T. Dumais, and D. J. Liebling, "Characterizing microblogs with topic models," in *Proc. Int. Conf. Weblogs Social Media*, 2010, pp. 130–137.

[16] D. Quercia, H. Askham, and J. Crowcroft, "TweetLDA: Supervised topic classification and link prediction in twitter," in *Proc. 3rd Annu. ACM Web Sci. Conf.*, 2012, pp. 247–250.

[17] J. Chang and D. M. Blei, "Hierarchical relational models for document networks," *The Ann. Appl. Statist.*, vol. 4, no. 1, pp. 124–150, 2010.

[18] P. Hennig, D. Stern, R. Herbrich, and T. Graepel, "Kernel topic models," in *Proc. 15th Int. Conf. Artif. Intell. Statist.*, 2012, pp. 1–9.

[19] J. Chuang, C. D. Manning, and J. Heer, "Without the clutter of unimportant words": Descriptive keyphrases for text visualization," *ACM Trans. Comput.-Human Interaction*, vol. 19, no. 3, pp. 1–29, 2012.

[20] J. Chuang, J. Heer, and C. D. Manning, "Termite: Visualization techniques for assessing textual topic models," in *Proc. Int. ACM Working Conf. Adv. Vis. Interfaces*, 2012, pp. 74–77.

[21] J. Chuang, D. Ramage, C. D. Manning, and J. Heer, "Interpretation and trust: Designing model-driven visualizations for text analysis," in *Proc. SIGCHI Conf. ACM Human Factors Comput. Syst.*, 2012, pp. 443–452.

[22] A. Schwaighofer, J. Q. Candela, T. Borchert, T. Graepel, and R. Herbrich, "Scalable clustering and keyword suggestion for online advertisements," in *Proc. 3rd Int. ACM Workshop Data Mining Audience Intell. Advertising*, 2009, pp. 27–36.

[23] Y. Chen, G.-R. Xue, and Y. Yu, "Advertising keyword suggestion based on concept hierarchy," in *Proc. 1st Int. Conf. Web Search Web Data Mining*, Feb. 2008, pp. 251–260.

[24] S. Ravi, A. Broder, E. Gabrilovich, V. Josifovski, S. Pandey, and B. Pang, "Automatic generation of bid phrases for online advertising," in *Proc. 3rd Int. ACM Conf. Web Search Data Mining*, Feb. 2010, pp. 341–350.

[25] S. Pandey, K. Punera, M. Fontoura, and V. Josifovski, "Estimating advertisability of tail queries for sponsored search," in *Proc. 33rd Int. ACM Conf. Res. Develop. Inform. Retrieval*, 2010, pp. 563–570.

[26] S. Thomaidou, K. Leymonis, and M. Vazirgiannis, "GrammAds: Keyword and ad creative generator for online advertising campaigns," in *Proc. 1st Int. Conf. Digit. Enterprise Des. Manag.*, 2013, vol. 205, pp. 33–44.

[27] A. Fujita, K. Ikushima, S. Sato, R. Kamite, K. Ishiyama, and O. Tamachi, "Automatic generation of listing ads by reusing promotional texts," in *Proc. 12th Int. ACM Conf. Electronic Commerce*, Aug. 2010, pp. 191–200.

[28] K. Liakopoulos, "Automatic advertising campaign development," *M.Sc. Thesis*, Athens Univ. Economics Bus., Athina, Greece, 2011.

[29] K. Hosanagar and V. Cherepanov, "Optimal bidding in stochastic budget constrained slot auctions," in *Proc. 9th ACM Conf. Electron. Commerce*, 2008, p. 20.

[30] R. Tibshirani, "Regression shrinkage and selection via the lasso," *J. Royal Statist. Soc. Series B*, vol. 58, no. 1, pp. 267–288, 1996.

[31] W. Fan, M. D. Gordon, and P. Pathak, "Effective profiling of consumer information retrieval needs: A unified framework and empirical comparison," *Dec. Support Syst.*, vol. 40, no. 2, pp. 213–233, 2004.

[32] S. Bird, E. Klein, and E. Loper, *Natural Language Processing with Python*. Sebastopol, CA, USA: O'Reilly Media, Inc., 2009.

[33] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *J. Mach. Learning Res.*, vol. 12, pp. 2825–2830, 2011.

[34] A. Inselberg, "Multidimensional detective," in *Proc. IEEE Symp. Inform. Vis.*, 1997, p. 100.

[35] M. Bostock, V. Ogievetsky, and J. Heer, "D3 data-driven documents," *IEEE Trans. Vis. Comput. Graph.*, vol. 17, no. 12, pp. 2301–2309, Dec. 2011.

Ahmet Bulut received the bachelor's degree in computer engineering from Bilkent University, and the PhD degree in computer science from UC Santa Barbara. He is an assistant professor of computer science at İstanbul Şehir University. He oversees Search Engine Marketing (SEM) operations at Udemy Inc. Previously, he managed SEM operations at Grou.ps Inc. He worked as a research scientist at Citrix Online, and as a senior researcher at Like.com. At Citrix Online, he was in the core set of people that architected and developed the first scalable and fault tolerant back-end infrastructure for the award winning online collaboration product GoToMeeting. He is a member of the IEEE.