Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2014, Article ID 818203, 9 pages
http://dx.doi.org/10.1155/2014/818203

Research Article

Predicting Click-Through Rates of New Advertisements Based on the Bayesian Network

Zhipeng Fang,[1] Kun Yue,[1,2] Jixian Zhang,[1] Dehai Zhang,[2,3] and Weiyi Liu[1]

[1] *Department of Computer Science and Engineering, School of Information Science and Engineering, Yunnan University, Kunming 650091, China*
[2] *Key Laboratory of Software Engineering of Yunnan Province, Kunming 650091, China*
[3] *Department of Software Engineering, School of Software, Yunnan University, Kunming 650091, China*

Correspondence should be addressed to Kun Yue; kyue@ynu.edu.cn

Received 23 April 2014; Accepted 12 July 2014; Published 24 July 2014

Academic Editor: Guangming Xie

Most classical search engines choose and rank advertisements (ads) based on their click-through rates (CTRs). To predict an ad's CTR, historical click information is frequently concerned. To accurately predict the CTR of the new ads is challenging and critical for real world applications, since we do not have plentiful historical data about these ads. Adopting Bayesian network (BN) as the effective framework for representing and inferring dependencies and uncertainties among variables, in this paper, we establish a BN-based model to predict the CTRs of new ads. First, we built a Bayesian network of the keywords that are used to describe the ads in a certain domain, called keyword BN and abbreviated as KBN. Second, we proposed an algorithm for approximate inferences of the KBN to find similar keywords with those that describe the new ads. Finally based on the similar keywords, we obtain the similar ads and then calculate the CTR of the new ad by using the CTRs of the ads that are similar with the new ad. Experimental results show the efficiency and accuracy of our method.

1. Introduction

Search engine has become an important means for finding information on the Internet today. Most classical search engines are funded through textual advertising placed next to their search results. Search engine advertising has become a significant element of the Web browsing experience [1]. The advertising revenue of the three classical search engines, Google, Yahoo, and Bing achieves at least 25 billion dollars per year and is still rising gradually [2]. Search engine advertising usually uses the keyword auction business model, and the advertisers pay cost of the keywords. The primary means is pay-per-click (PPC) with a cost-per-click (CPC) billing, which means that the search engine is paid every time the ad is clicked by users. An ad's popularity can be measured by its CTR, by which the ads on a search engine are generally ranked. The expected revenue from an ad is a function of CTR and CPC: CTR × CPC. The probability that a user clicks on an ad declines rapidly, as much as 90%, with the display position. Choosing the right ads for the query and the order in which they are displayed greatly affects the probability that a user will see and click on each ad [1]. So, this rank has a strong impact on the revenue that a search engine receives from ads. Further, showing users ads that they prefer to click on can improve users' satisfaction. Thus, accurately predicting the CTR of an ad is critical for maximizing the revenue and improving user satisfaction.

In recent years, CTR prediction has been widely concerned in academic communities of computational advertisement. For example, Agarwal et al. [3] divided the CTR prediction process into two stages. First, keywords are predefined as conceptual levels, and then the CTR for each different area is calculated in each level. Chakrabarti et al. [4] predicted the CTR based on the logistic regression and employed a multiplicative factorization to model the interaction effects for several regions. Regelson and Fain [5] predicted the CTR based on term level and adopted hierarchical clusters for the low frequency or completely novel terms. Dembczynski

et al. [6] predicted the CTR in view of decision rules. Xiong et al. [7] designed a model based on the continuous conditional random fields and considered both features of an ad and its similarity to the surrounding ones. Actually, considering the accurate recommendation of ads in line with their characteristics, CTR prediction is dependent on both users and advertisements. Meanwhile, from the behavioral targeting point of view, CTR can be increased by obtaining the users' favors pertinently, which were also intensively studied recently [8–11]. Wang and Chen [12] developed several machine learning algorithms, including conditional random fields, support vector machines, decision tree, and back propagation neural networks, to learn users' behaviors from searching and clicking logs in order to predict the CTR.

In general, the above methods are only suitable for the ads that have plentiful historical click logs, except for the new ads (without plentiful historical click logs). It is known that all advertising impressions and clicks follow a mathematical relationship known as power-law distribution [3], and the search keyword frequency follows the power-law distribution as well [5]. This means that a large number of keywords will be concerned only in a small number of search behaviors in a given period of time, and a small number of search behaviors will imply a small number of potential impressions. Thus, the CTRs of a large number of new ads are unknown, while the prediction is not trivial since we do not have plentiful historical data about these ads. This is exactly the focus of this paper.

It is natural to consider the uncertainties in ads and CTR prediction, while uncertainty related mechanisms actually have been incorporated into CTR prediction in recent years [13–15]. For example, Graepel et al. [2] presented a dichotomy method to predict the sponsored search advertising CTR and an online Bayesian probability regression algorithm. Chapelle and Zhang [16] proposed the model for users' browsing behavior to make CTR prediction. However, the above methods cannot be well suitable for CTR prediction of new ads. Fortunately, it is well known that Bayesian network (BN) is an effective framework of representing and inferring uncertainties among random variables [17]. A BN is a directed acyclic graph (DAG), where nodes represent random variables and edges represent dependencies among these random variables. Each variable in a BN is associated with a conditional probability table (CPT) to give the probability of each parent state. By means of BN-based probabilistic inferences, the unknown dependencies can be obtained under the current situations. Thus, in this paper, we adopt BN as the underlying framework for predicting the CTRs of new ads. For this purpose, we will have to address the following two problems: (1) constructing a BN from the keywords that are used to describe the ads, (2) providing an efficient inference mechanism to predict the probability distributions of all possible values for the given keywords that are used to describe the new ad.

To construct the BN from the keywords that are used to describe the ads, we first find out the keywords appearing in the user queries and the ads. If the same keyword appears in the user queries and the ads simultaneously, then the ads associated with this keyword may be clicked by users. So, we use these keywords as BN's nodes, where the edges between nodes are to describe the relationship between similar keywords. The constructed BN is called keyword BN, abbreviated as KBN.

To predict the CTR of the new ad, we could use similar ads' CTRs. In order to obtain similar ads, we use the probabilistic inferences on the KBN to obtain the similar keywords at first. This makes the BN be reasonably looked upon as the underlying model of probabilistic inferences for predicting the CTRs of new ads. Many algorithms for BNs exact inferences have been proposed [18], but these methods are run in exponential time, which are not efficient and suitable enough with respect to the BN-based inferences, especially over large scale keyword BNs. Thus, based on Gibbs sampling [18], we propose an approximate inference algorithm of the KBN for obtaining the probability distribution to find similar keywords and ultimately predict the new ad's CTR.

Generally, the main contributions of this paper are as follows.

(1) We propose an efficient method to construct the KBN from the keywords to describe the given ads, as the basis of probabilistic inferences and CTR prediction.

(2) We propose an algorithm for KBN's approximate inferences to predict the probability distributions of possible values for the known keywords and correspondingly give the idea to predict the new ad's CTR.

(3) We implement the proposed algorithms and make preliminary experiments to test the feasibility of our method.

The remainder of this paper is organized as follows. In Section 2, we construct a KBN from the given keywords. In Section 3, we infer the probability distributions of the known keywords and then predict the new ad's CTR. In Section 4, we show experimental results and performance studies. In Section 5, we conclude and discuss future work.

2. Keyword Similarity Model

2.1. Basic Idea and Definitions. Search engine advertising usually uses keywords to describe ads, and advertisers pay for the cost of these keywords. Thus, in this paper, we use the set of keywords to describe ads and user queries, defined as follows.

Definition 1. Let $A = \{A_1, A_2, \ldots, A_m\}$, $Q = \{q_1, q_2, \ldots, q_l\}$, $K_A = \{K_1, K_2, \ldots, K_m\}$, and $K_N = \{k_{n1}, k_{n2}, \ldots, k_{nj}\}$ denote the set of the known ads, the query words, the known ads' keywords, and the new ad's keywords, respectively. Let $K_i = \{k_{i1}, k_{i2}, \ldots, k_{il}\}$ denote the set of keywords of ad A_i.

If the same keyword k appears in K_A and Q simultaneously, then the ads associated with k may be clicked by users. To depict the associations among A, Q, and K_A, we focus on all the keywords like k, denoted as K_x. As mentioned in Section 1, we adopt the graphical model as the preliminary

framework, where nodes correspond to the keywords in K_x and

$$K_x = Q \cap K_A. \tag{1}$$

Actually, (1) describes the cooccurrence of the keywords in the user queries and the ads whose CTRs are unknown (i.e., are to be predicted). This means that K_x reflects the inherent functionality or sense that may be requested by users. We look upon this as the similarity of keywords. To establish the model for globally representing the qualitative and quantitative similarity among keywords, as well as the uncertainty of the similarity, we first describe the basic concept of BN as follows.

A BN is a DAG $G = (V, E)$, in which the following hold [19].

(1) V is the set of random variables and makes up the nodes of the network.

(2) E is the set of directed edges connecting pairs of nodes. An arrow from node X to node Y means that X has a direct influence on Y ($X, Y \in V$ and $X \neq Y$).

(3) Each node has a CPT that quantifies the effects that the parents have on the node. The parents of node X are all those that have arrows pointing to it.

Based on the definition of a general BN, following we give the definition of keyword Bayesian network. Formally, a KBN is a pair $G = (G_K, S)$, defined as follows.

Definition 2. We use a pair $G = (G_K, S)$ to define the keyword Bayesian network (abbreviated as KBN) as follows.

(1) $G_K = (K_x, E)$ is the structure of the KBN, where $K_x = \{k_1, k_2, k_3, \ldots, k_x\}$ is a set of nodes in G_K. Each node in the DAG corresponds to a keyword, and each node has two values (1 and 0). The value 1 indicates that the keyword is in the keyword set of ad A_x, and the value 0 indicates that the keyword is not in the keyword set of ad A_x. If $k_j \rightarrow k_i$ holds, then we define k_j as a parent node of k_i, and we use $P_a(k_i)$ to represent the set of parent nodes of k_i.

(2) S is a set of conditional probabilities and consists of each node's CPT.

2.2. Constructing KBN from Ads' Keywords. To construct the KBN from the given keywords is to construct the DAG and calculate each node's CPT. Without loss of generality, the critical and difficult step in KBN construction is to construct the DAG, which is consistent with that of general BN's construction [18]. Then, we can calculate each node's CPT easily based on the constructed DAG.

The set of directed edges connecting pairs of nodes in the KBN describes the relationship between similar keywords. This means that, to describe the relationship between similar keywords, we will have to address the following two problems: (1) whether there is similar relationship between keywords, namely, whether there is an edge between the corresponding two nodes; (2) how to determine the edge's direction.

For problem (1), we consider the ratio of the number of ads that contain the specific two keywords and the number of ads that contain at least one of them. The direct similarity of the two keywords is established upon the above ratio. With the increase of the ratio values, the similarity between the two keywords will be increased correspondingly. If the ratio is higher than a certain threshold, we can regard that there is an undirected edge between these two keywords. Therefore, we use $\mathrm{sim}(k_i, k_j)$ to define the relationship strength between keywords k_i and k_j as follows:

$$\mathrm{sim}\left(k_i, k_j\right) = \frac{N\left(k_i = 1, k_j = 1\right)}{N\left(k_i = 1\right) + N\left(k_j = 1\right) - N\left(k_i = 1, k_j = 1\right)}, \tag{2}$$

where $N(k_i = 1, k_j = 1)$ denotes the number of ads that contain the keywords k_i and k_j. $N(k_i = 1) + N(k_j = 1) - N(k_i = 1, k_j = 1)$ denotes the ads that contain the keywords k_i or k_j.

Let ε be the given similarity threshold, which could be determined based on empirical tests or specific experience. If $\mathrm{sim}(k_i, k_j) > \varepsilon$, we believe that k_i and k_j are similar. That is, there should be an undirected edge between the nodes corresponding to these two keywords, respectively.

For problem (2), we consider any two nodes that are connected by an edge. We can compare the probability of the case that k_i will also appear when k_j appears, denoted as $P(k_i \mid k_j)$, as well as the opposite case, denoted as $P(k_j \mid k_i)$. Then, we can compare the probability of k_i's appearance on that of k_j and the probability of k_j's appearance on the that of k_i. This makes it possible to obtain the direction of the edge between k_i and k_j, which reflects the dominate keyword concerned in the mutual dependence between these two keywords. Specifically, $P(k_j \mid k_i)$ and $P(k_j \mid k_i)$ can be calculated as follows:

$$\begin{aligned} P\left(k_i \mid k_j\right) &= \frac{N\left(k_i = 1 \mid k_j = 1\right)}{N\left(k_j = 1\right)}, \\ P\left(k_j \mid k_i\right) &= \frac{N\left(k_j = 1 \mid k_i = 1\right)}{N\left(k_i = 1\right)}, \end{aligned} \tag{3}$$

where $N(k_i = 1 \mid k_j = 1)$ means the number of ads that contain k_i when the ads contain k_j. $N(k_j = 1)$ means the number of ads containing k_j.

So, we can compare $P(k_i \mid k_j)$ and $P(k_j \mid k_i)$. If $P(k_i \mid k_j) > P(k_j \mid k_i)$, then the edge should be directed from k_j to k_i, since the dependency of k_i on k_j is larger than that of k_j on k_i. According to the above discussion, we summarize the method for KBN construction in Algorithm 1.

It should be noted that no cycles will be generated when executing Algorithm 1, since the direction of the edge between k_i and k_j is dependent on $P(k_i \mid k_j)$ and $P(k_j \mid k_i)$, where $P(k_i \mid k_j) > P(k_j \mid k_i)$ or exclusively $P(k_j \mid k_i) > P(k_i \mid k_j)$ holds. We consider a DAG with n nodes; the similarity computations and direction decisions (Step 5~Step 9) will be done for $O(n^2)$ times. Example 3 illustrates the execution of Algorithm 1. Likelihood estimation [18] is commonly used to

Input: $Q = \{q_1, q_2, \ldots, q_l\}$, a set of query;
$K_A = \{K_1, K_2, \ldots, K_m\}$, a set of the known ads' keywords;
ε, threshold of the similarity of two keywords ($0 \le \varepsilon \le 1$)
Output: $G = (V, E)$, DAG of the KBN
Variable: n, the number of nodes in the KBN
Steps:
(1) $K_x \leftarrow Q \cap K_A$
(2) $V \leftarrow K_x, E \leftarrow \{\}$
(3) For $i \leftarrow 1$ to $n - 1$ Do
(4) For $j \leftarrow i + 1$ to n Do
(5) If $\text{sim}(k_i, k_j) > \varepsilon$ Then // By (2)
(6) If $P(k_i \mid k_j) \ge P(k_j \mid k_i)$ Then $E \leftarrow E \cup \{kj \rightarrow k_i\}$ // By (3)
(7) Else $E \leftarrow E \cup \{k_i \rightarrow k_j\}$
(8) End If
(9) End If
(10) End For
(11) End For
(12) Return G

Algorithm 1: KBN construction.

estimate the parameters of a statistical method by counting the frequency from tuples. We adopt this method to compute the CPT for each variable easily upon the KBN structure, where the probability is

$$P(k_i \mid Pa(k_i)) = \frac{P(k_i, Pa(k_i))}{P(Pa(k_i))} = \frac{\text{the number of the ads that contain } k_i \text{ and } Pa(k_i)}{\text{the number of the ads that contain } Pa(k_i)}. \quad (4)$$

It is worth noting that the execution of computing CPTs in the KBN by (4) is dependent on the executing time of querying the sample data. This means that (4) can be evaluated in different consuming times under different mechanisms of storage and query processing of the sample data. In particular, we adopted MongoDB in our implementation, and the efficiency of CPTs computation will be tested by experiments, given in Section 4.

Example 3. Let $Q = \{k_1, k_2, k_3, k_4\}$, $K_A = \{K_1 = \{k_1, k_2, k_6\}$, $K_2 = \{k_1, k_2, k_5\}$, $K_3 = \{k_1, k_2, k_3, k_4\}$, $K_4 = \{k_1, k_2, k_3, k_4\}$, $K_5 = \{k_1, k_3, k_6\}$, $K_6 = \{k_3, k_4\}$, and $K_7 = \{k_2, k_4\}\}$ be user queries and the keywords of the ads with known CTRs.

By Step 1 in Algorithm 1, we compare the set Q and the set K_A to get the nodes in the DAG, k_1, k_2, k_3, and k_4.

By Step 5 in Algorithm 1, we calculate the similarity between each two keywords based on (2), such as $\text{sim}(k_1, k_2) = 0.667$, $\text{sim}(k_2, k_3) = 0.286$, and $\text{sim}(k_3, k_4) = 0.6$. Suppose that ε is 0.40, and then there are four edges (k_1, k_2), (k_1, k_3), (k_2, k_4), and (k_3, k_4) that can be added into E.

By Step 6 in Algorithm 1, we determine each edge's direction based on (3), such as $P(k_2 \mid k_4) = 0.75$, $P(k_4 \mid k_2) = 0.6$. So, the edge between k_2 and k_4 should be directed from k_4 to k_2. Upon the KBN structure, the CPTs can be computed by (4). Finally we can obtain the KBN shown in Figure 1.

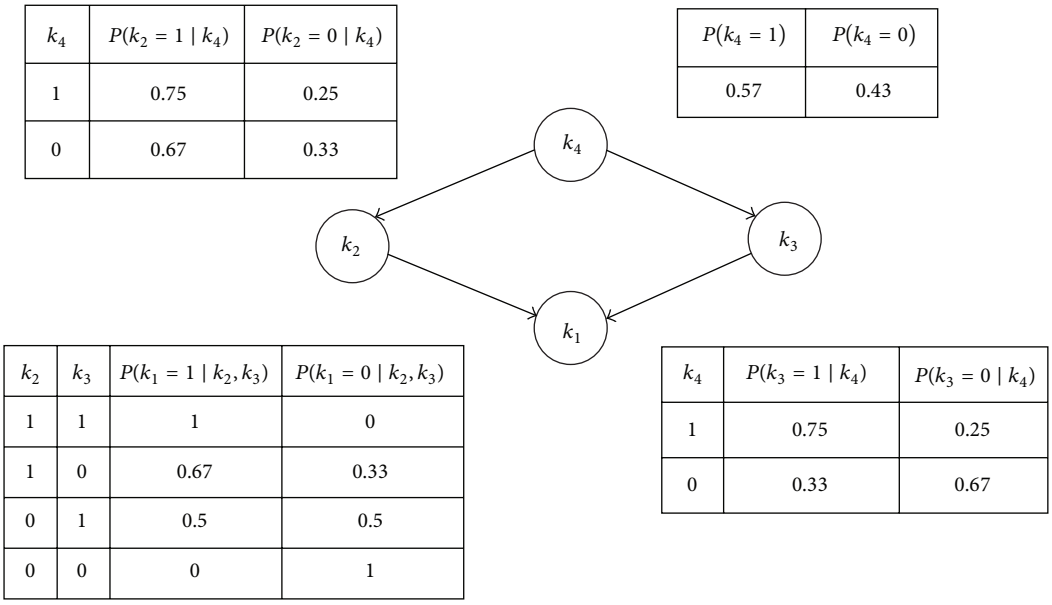
3. Probabilistic Inferences of KBN and CTR Prediction

3.1. Approximate Inferences of KBN. In order to predict the CTR of a new ad, we are to find the keywords of the ads with known CTRs, which are similar to the keywords of the new ad. Thus, the similarity of ads' functionality or sense can be inferred and discovered. Consequently, we can use the similarity between the keywords of the new ad and those of the ads with known CTRs to find the ads that are related to the new ad.

Although we can obtain the keywords with the direct similarity relationship by the ideas presented in Section 2.2, there are still many desirable indirect similarity relationships between different keywords, which is also the motivation of deriving the KBN. Thus, we make use of the KBN inferences to find the indirect similarity relationships between the keywords of the new ad and those of the ad with known CTRs. For this purpose, we can calculate the conditional probability of the keyword (KBN node) of the new ad in the case of the given keywords as evidence in KBN inferences.

It is known that Gibbs sampling is a Markov chain Monte Carlo algorithm and it always generates a Markov chain of samples [20–22]. Gibbs sampling is particularly well-adapted to sampling the posterior distributions of a BN [23]. To infer the probability distributions of all possible values, we extend the Gibbs sampling algorithm to the KBN inference and give the approximate inference method in Algorithm 2. The basic idea is given as follows.

(1) We generate a random sample consistent with the evidence nodes. In the KBN, the evidence nodes are the new ad's keywords, where we assume that the new



k_4	$P(k_2 = 1 \mid k_4)$	$P(k_2 = 0 \mid k_4)$
1	0.75	0.25
0	0.67	0.33

$P(k_4 = 1)$	$P(k_4 = 0)$
0.57	0.43

k_2	k_3	$P(k_1 = 1 \mid k_2, k_3)$	$P(k_1 = 0 \mid k_2, k_3)$
1	1	1	0
1	0	0.67	0.33
0	1	0.5	0.5
0	0	0	1

k_4	$P(k_3 = 1 \mid k_4)$	$P(k_3 = 0 \mid k_4)$
1	0.75	0.25
0	0.33	0.67

Figure 1: A simple KBN.

Input: $G = (K, E)$, the DAG of the KBN;
K_B: the evidence nodes in G (the new ad's keywords);
Z: the non-evidence nodes in G (all keywords except for the new ad's keywords);
e: the set of values of the evidence (set 1 as the value);
q: the query variable (the keyword will be inferred);
s: the total times of sampling.
Output: The estimates of $P(q = 1 \mid e)$
Variables:
z: the set of the values of the non-evidence;
$k_{(-i)} = \{k_1, k_2, \ldots, k_{i-1}, k_{i+1}, \ldots, k_g\}$;
$v = e \cup z$: the current state of the KBN;
$N(q_1)$: the number of the samples when the value of q is 1
Steps:
(1) Initialization:
$z \leftarrow$ random values of k_i in Z
$v^{(0)} \leftarrow e \cup z, c^{(0)} \leftarrow v^{(0)}$
$N(q_1) \leftarrow 0$
(2) For $m \leftarrow 1$ to s Do
(i) Compute the probabilities of the selected variables based on the state $c^{(m-1)}$,
Randomly select one non-evidence variable k_i from Z:
$B \leftarrow P(k_i = 0 \mid c_{MB(k_i)}) + P(k_i = 1 \mid c_{MB(k_i)})$, where $c_{MB(k_i)}$ is the set of the values in the Markov chain of k_i in the current state c
(ii) Generate a random number $r_k \in [0, B]$ and we determine the value of k_i:

$$k_i = \begin{cases} 0 & r_k \le P\left(k_i = 0 \mid c_{MB(k_i)}\right) \\ 1 & P\left(k_i = 0 \mid c_{MB(k_i)}\right) < r_k \le P\left(k_i = 0 \mid c_{MB(k_i)}\right) + P\left(k_i = 1 \mid c_{MB(k_i)}\right) \end{cases} \quad (5)$$

$v^{(k)} \leftarrow \left(v_{(-i)}{}^{(k-1)}, k_i\right), c^{(k)} \leftarrow v^{(k)}$
(iii) Count
If $c_q{}^{(k)} = 1$ Then $N(q_1) \leftarrow N(q_1) + 1$
End For
(3) Estimate $P(q = 1 \mid e)$
$P(q = 1 \mid e) \leftarrow N(q_1)/s$

Algorithm 2: KBN inference.

ad's keywords are in the KBN. Then, for the query expressed as $P(q = 1 \mid e)$, where q is the target (i.e., keyword in the new ad) and e is the evidence.

(2) Nonevidence nodes are sampled randomly, including all the keywords in the KBN except the new ad's keywords. From the CPTs of the KBN, the conditional probabilities of nonevidence nodes can be obtained given their Markov chains, respectively. Thus, the new state can be achieved. This process will be iterated until the given threshold sampling time is reached.

(3) A set of samples can be generated, and the corresponding probability distributions can be achieved. The desired probabilities of $P(q = 1 \mid e)$ can be obtained as the answer to the given query.

By Algorithm 2, we can obtain the keywords with direct or indirect similarity relationships. If $P(k_i = 1 \mid k_j = 1) > \lambda$, then we regard that the keyword k_i is similar to k_j. For a KBN with n nodes, the invocation times of KBN inferences of Algorithm 2 will be $O(n^2)$. It was concluded that [18, 23, 24], as long as the sampling times are many enough, the Gibbs sampling always converges to a stationary distribution. Particularly, Russell et al. [18] have given the following conclusion to guarantee that the estimate of posterior probabilities returned by a Gibbs sampling algorithm will converge to the true answer if the total sampling times are large enough, stated as Theorem 4, which also holds for Algorithm 2 given in this section when inferring the KBN.

Let $P(\vec{q} \rightarrow \vec{q}')$ be the probability that the process makes a transition from state $\vec{q}$ to $\vec{q}'$. Let $\pi_k(\vec{q})$ be the probability in state $\vec{q}$ at time k and let $\pi_{k+1}(\vec{q}')$ be that in state $\vec{q}'$ at time $k+1$. Given $\pi_k(\vec{q})$, we can compute $\pi_{k+1}(\vec{q}')$ by summing:

$$\pi_{k+1}\left(\vec{q}'\right) = \sum_{\vec{q}} \pi_k\left(\vec{q}\right) P\left(\vec{q} \longrightarrow \vec{q}'\right). \tag{5}$$

We say the process has reached its stationary distribution if $\pi_k = \pi_{k+1}$, which means that the sampling process is converged and $\pi_{k+1}(\vec{q}') = \sum_{\vec{q}} \pi_k(\vec{q}) P(\vec{q} \rightarrow \vec{q}')$ for all $\vec{q}'$.

Theorem 4. *If the following equation holds, then* $\pi_{k+1}(\vec{q}') = \sum_{\vec{q}} \pi_k(\vec{q}) P(\vec{q} \rightarrow \vec{q}')$*:*

$$\pi\left(\vec{q}\right) P\left(\vec{q} \longrightarrow \vec{q}'\right) = \pi\left(\vec{q}'\right) P\left(\vec{q}' \longrightarrow \vec{q}\right) \quad \forall \vec{q},\ \vec{q}'. \tag{6}$$

This guarantees the convergence and effectiveness of Algorithm 2 theoretically. Following, we illustrate the execution of Algorithm 2 by an example.

Example 5. We consider the query $P(k_4 = 1 \mid k_1 = 1)$ applied to the KBN in Figure 1.

By Step 1 in Algorithm 2, the nonevidence nodes k_2 and k_3 are initialized randomly. We suppose the current state is $[k_1 = 1, k_2 = 0, k_3 = 0, k_4 = 1]$. The following steps are executed repeatedly.

By Step 2 in Algorithm 2, k_2 is sampled given the current state of its Markov blanket variables. In this case, we can obtain the probabilities when k_2 is sampled as 1 and 0, respectively, $P(k_2 = 1 \mid k_1 = 1, k_3 = 0, k_4 = 1) = 1$ and $P(k_2 = 0 \mid k_1 = 1, k_3 = 0, k_4 = 1) = 0$.

If we suppose $r_2 = 0.50$, then the result will be 1 and the new state will be $[k_1 = 1, k_2 = 1, k_3 = 0, k_4 = 1]$. k_3 is sampled, given the current state of its Markov blanket variables. In this case, we can obtain the probabilities when k_3 is sampled as 1 and 0, respectively, $P(k_3 = 1 \mid k_1 = 1, k_2 = 1, k_4 = 1) = 0.82$, $P(k_3 = 0 \mid k_1 = 1, k_2 = 1, k_4 = 1) = 0.18$. If we suppose that $r_3 = 0.10$, then the sampling result will be 0 and the new state will be $[k_1 = 1, k_2 = 1, k_3 = 0, k_4 = 1]$ as well.

By Step 3 in Algorithm 2, if the process visits 60 states of $k_4 = 0$ and 140 states of $k_4 = 1$, then we can obtain $P(k_4 = 1 \mid k_1 = 1) = 0.7$, which is the answer to the given query.

3.2. KBN-Based CTR Prediction. Based on the results of KBN inferences given in Section 3.1, we can obtain the set of similar ads described by the similar keywords. Consequently, we can predict the CTR of the new ad by using the similar ads' CTRs that are known already. Let $A = \{A_1, A_2, \ldots, A_i\}$ denote the set of similar ads to the target new ad, and $C = \{C_1, C_2, \ldots, C_i\}$ denote the corresponding CTRs of A. The CTR of the new ad can be calculated as follows:

$$C_N = \frac{\left(\sum_{n=1}^{i} C_n\right)}{i}. \tag{7}$$

This means that we simply use the average CTR of the similar ads (with known CTRs) as the prediction of the new ad's CTR. The idea is illustrated by the following example.

Example 6. Suppose k_2, k_3, and k_4 are similar keywords of k_1 by Algorithm 2. If the new ad has only one keyword k_1, then we can find the similar ads A_3 and A_4 that contain k_1, k_2, k_3, and k_4. So, we can use the CTRs of A_3 and A_4 to predict the CTR of the new ad by (7), and we have $C_N = (C_3 + C_4)/2$.

4. Experimental Results

To test the feasibility of the ideas proposed in this paper, we implemented our methods for constructing and inferring KBN, as well as that for predicting the new ad's CTR. We mainly tested the efficiency of KBN construction, the efficiency, correctness, and convergence of KBN inference, and the accuracy of the KBN-based CTR prediction.

In the experiments, we adopted the test data from KDD Cup 2012-Track 2 [25]. All the data sets were stored in MongoDB and all the codes were written in Python 2.7. The machine configurations are as follows: Pentium(R) Dual-Core CPU E5700 @3.00 GHz with 2 GB main memory, running Ubuntu 12.04 64-bit operating system. According to (1), we merged all the ads' keywords and then obtained 27512 keywords in total as the original test dataset.

4.1. Efficiency of KBN Construction. We chose $10, 20, \ldots, 100$ keywords from the original dataset and tested the execution

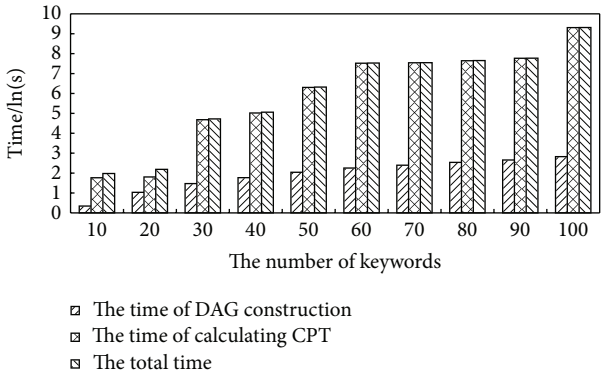


Figure 2: Efficiency of KBN construction.

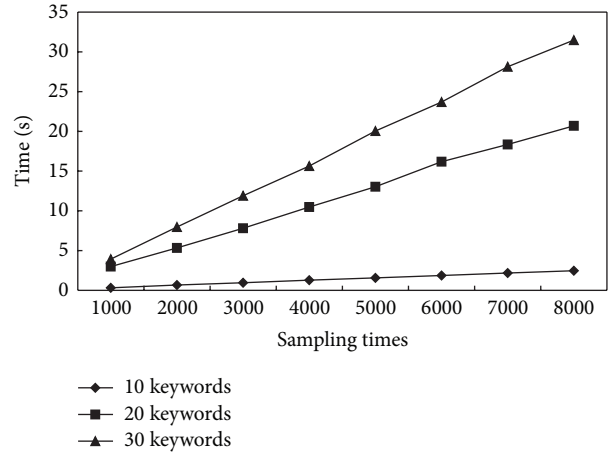


Figure 3: Efficiency of KBN inference.

time of KBN construction. For each KBN, we recorded the average time of 10 times of tests. The execution time of DAG construction, CPT calculation, and the total time were recorded and shown in Figure 2 by logarithm scale. It can be seen that the execution time of DAG construction is much smaller than that of CPT calculation, and the total time mainly depends on that of CPT calculation. This is consistent with the actual situations, since the CPT calculation of each node concerns many good statistical computations (i.e., aggregation queries on MongoDB), which are exponential to the parent number of this node. The above observation makes our method for KBN construction be feasible with respect to small scales of keywords under the hardware configurations like our experimental environment. Basically, the execution time of KBN construction is increased lineally with the increase of the number of the nodes when the number of keywords is not larger than 100.

Thus, from the efficiency point of view, the above observations make our further improvement of KBN construction be feasible by incorporating data-intensive computing techniques for the aggregation query processing. This is exactly our future work.

4.2. Precision, Efficiency, and Convergence of KBN Inference. To test the precision of the KBN inference, we established the tests on the KBN in Example 3. For all possible specific values of the same evidence and target variables in the KBN in Figure 1, we compared the inference results of the KBN obtained by Algorithm 2 and those obtained by Netica [26] that were adopted as the criteria of the precision of KBN inferences. The errors of KBN inferences were defined as the difference between the above two inference results for each pair of evidence and target. The inference results based on Netica, those based on Algorithm 2, and the corresponding errors are shown in Table 1. It can be seen that the maximal, minimal, and average errors are 10.9%, 7.8%, and 9.15%, respectively, which can be accepted generally and looked upon as precise basically. This also means that the KBN inference results are precise to a certain extent and thus well verifies the applicability and feasibility of KBN and its approximate inference algorithm for finding similar keywords.

Table 1: Errors of KBN inference.

	Netica (%)	KBN (%)	Error (%)
$P(k_1 = 1 \mid k_3 = 1)$	91.1	83.3	7.8
$P(k_1 = 0 \mid k_3 = 1)$	8.91	16.7	7.8
$P(k_2 = 1 \mid k_3 = 1)$	73.0	83.2	10.2
$P(k_2 = 0 \mid k_3 = 1)$	27.0	16.8	10.2
$P(k_4 = 1 \mid k_3 = 1)$	75.1	83.9	8.8
$P(k_4 = 0 \mid k_3 = 1)$	24.9	16.1	8.8
$P(k_1 = 1 \mid k_3 = 0)$	34.8	23.9	10.9
$P(k_1 = 0 \mid k_3 = 0)$	65.2	76.1	10.9
$P(k_2 = 1 \mid k_3 = 0)$	69.6	77.9	8.3
$P(k_2 = 0 \mid k_3 = 0)$	30.4	22.1	8.3
$P(k_4 = 1 \mid k_3 = 0)$	33.1	24.2	8.9
$P(k_4 = 0 \mid k_3 = 0)$	66.9	75.8	8.9

To test the efficiency and convergence of Algorithm 2, we adopted the KBN constructed from the test dataset. With the increase of the number of sampling times, we recorded the execution time of Algorithm 2 and the inference results under different number of keywords, shown in Figures 3 and 4, respectively. It can be seen from Figure 3 that the execution time of Algorithm 2 was increased linearly, and the increase of execution time with the increase of keywords under the same sample size is not sensitive as well. It can be seen from Figure 4 that the inference results converge to a stable value rapidly after less than 4000 iterations of sampling. The observations from Figures 3 and 4 show that Algorithm 2 for KBN inferences is scalable and efficient to a certain extent.

4.3. Accuracy of CTR Prediction. We randomly selected 500 ads from the test dataset, and we compared the ads' CTRs predicted by (7) in Section 3.2 and the real CTRs of these ads, where each ad's real CTR was evaluated by the following well-adopted metric: click/impression (i.e., the average click for each impression). To test the accuracy of the CTR prediction, we selected 10 ads, for each of which we recorded the ad ID, real CTR (denoted as rCTR), predicted CTR (denoted as pCTR), and the difference between rCTR and pCTR (denoted

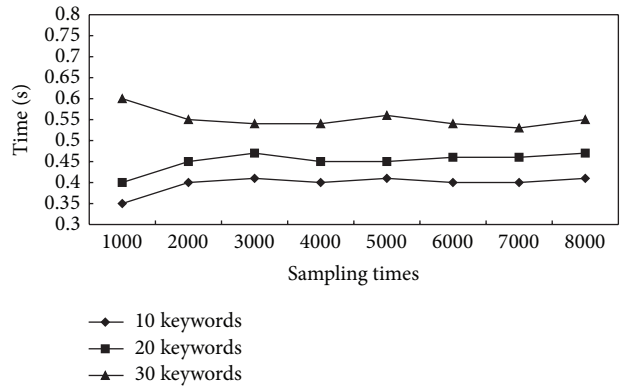


Figure 4: Convergence of KBN inference.

Table 2: Correctness of CTR prediction.

Ad ID	rCTR (%)	pCTR (%)	Error (%)
21442606	42.2	53.1	10.9
10950981	42	64	22
20302019	70.5	71.2	0.7
20158057	100	77	23
10757736	33.3	67	33.7
10110478	62.5	77	14.5
10228933	100	46	54
20174702	73.6	65.6	8
20277194	50	79	29
20745473	50	46	4

as Error), shown as in Table 2. We can obtain the maximum, minimum, and the average error of the predicted CTRs as 54%, 0.7%, and 19.6%, respectively. Thus, we can conclude that the method for CTR prediction proposed in this paper can be used to predict the new ad's CTR accurately on average. Accordingly, to improve the general accuracy of the CTR prediction is exactly also our future work.

5. Conclusions and Future Work

Predicting CTRs for new ads is extremely important and very challenging in the field of computational advertising. In this paper, we proposed an approach for predicting the CTRs of new ads by using the other ads with known CTRs and the inherent similarity of their keywords. The similarity of the ad keywords establishes the similarity of the semantics or functionality of the ads. Adopting BN as the framework for representing and inferring the association and uncertainty, we mainly proposed the methods for constructing and inferring the keyword Bayesian network. Theoretical and experimental results verify the feasibility of our methods.

To make our methods applicable in realistic situations, we will incorporate the data-intensive computing techniques to improve the efficiency of the aggregation query processing when constructing KBN from the large scale test dataset. Meanwhile, we will also improve the performance of KBN inferences and the corresponding CTR prediction. In this paper, we assume that all the new ad's keywords are included in the KBN, which is the basis for exploring the method for the situation if not all the keywords are not included (i.e., some keywords of the new ads are missing). Moreover, we can also further explore the accurate user targeting and CTR prediction based on the ideas given in this paper. These are exactly our future work.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This paper was supported by the National Natural Science Foundation of China (nos. 61163003, 61263043), the Yunnan Provincial Foundation for Leaders of Disciplines in Science and Technology (no. 2012HB004), the Natural Science Foundation of Yunnan Province (nos. 2011FB020, 2013FB010), and the Research Foundation of Key Laboratory of Software Engineering of Yunnan Province (no. 2012SE013).

References

[1] M. Richardson, E. Dominowska, and R. Ragno, "Predicting clicks: estimating the click-through rate for new ads," in *Proceedings of the 16th International World Wide Web Conference (WWW '07)*, pp. 521–530, May 2007.

[2] 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 (ICML '10)*, pp. 13–20, Haifa, Israel, June 2010.

[3] D. Agarwal, A. Z. Broder, D. Chakrabarti, D. Diklic, V. Josifovski, and M. Sayyadian, "Estimating rates of rare events at multiple resolutions," in *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '07)*, pp. 16–25, August 2007.

[4] D. Chakrabarti, D. Agarwal, and V. Josifovski, "Contextual advertising by combining relevance with click feedback," in *Proceeding of the 17th International Conference on World Wide Web (WWW '08)*, pp. 417–426, New York, NY, USA, April 2008.

[5] M. Regelson and D. Fain, "Predicting click-through rate using keyword clusters," in *Proceedings of the 2nd Workshop on Sponsored Search Auctions*, 2006.

[6] K. Dembczynski, W. Kotlowski, and D. Weiss, "Predicting ads' click-through rate with decision rules," in *Proceedings of the Workshop on Targeting and Ranking in Online Advertising (WTROA '08)*, pp. 21–25, 2008.

[7] C. Xiong, T. Wang, W. Ding, Y. Shen, and T. Liu, "Relational click prediction for sponsored search," in *Proceedings of the 5th ACM International Conference on Web Search and Data Mining (WSDM '12)*, pp. 493–502, February 2012.

[8] S. Gollapudi, R. Panigrahy, and M. Goldszmidt, "Inferring click-through rates on ads from click behavior on search results," in *Proceedings of the Workshop on User Modeling for Web Applications (WSDM '11)*, Hong Kong, China, February 2011.

[9] J. Yan, N. Liu, G. Wang, W. Zhang, Y. Jiang, and Z. Chen, "How much can behavioral targeting help online advertising?" in *Proceedings of the 18th International World Wide Web Conference (WWW '09)*, pp. 261–270, April 2009.

[10] A. Ahmed, Y. Low, M. Aly, V. Josifovski, and A. J. Smola, "Scalable distributed inference of dynamic user interests for behavioral targeting," in *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '11)*, pp. 114–122, August 2011.

[11] X. Wang, W. Li, Y. Cui, R. Zhang, and J. Mao, "Click-through rate estimation for rare events in online advertising," in *Online Multimedia Advertising: Techniques and Technologies*, pp. 1–12, 2011.

[12] C. Wang and H. Chen, "Learning user behaviors for advertisements click prediction," in *Proceedings of the 34th Annual ACM SIGIR Conference (SIGIR '11)*, pp. 1–6, Beijing, China, 2011.

[13] F. Guo, C. Liu, A. Kannan et al., "Click chain model in web search," in *Proceedings of the 18th International World Wide Web Conference (WWW '09)*, pp. 11–20, April 2009.

[14] W. Chen, Z. Ji, S. Shen, and Q. Yang, "A whole page click model to better interpret search engine click data," in *Proceedings of the 25th Conference on Artificial Intelligence (AAAI '11)*, 2011.

[15] C. Liu, F. Guo, and C. Faloutsos, "BBM: Bayesian browsing model from petabyte-scale data," in *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '09)*, pp. 537–545, July 2009.

[16] O. Chapelle and Y. Zhang, "A dynamic Bayesian network click model for web search ranking," in *Proceedings of the 18th International World Wide Web Conference (WWW '09)*, pp. 1–10, April 2009.

[17] A. Darwiche, *Modeling and Reasoning with Bayesian Networks*, Cambridge University Press, 2009.

[18] S. Russell, P. Norvig, J. Canny, J. Malik, and D. Edwards, *Artificial Intelligence: A Modern Approach*, Prentice Hall, Englewood Cliffs, NJ, USA, 3rd edition, 1995.

[19] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, Morgan Kaufmann, Boston, Mass, USA, 1988.

[20] D. Spiegelhalter, A. Thomas, N. Best, W. Gilks, and D. Lunn, *BUGS: Bayesian inference using Gibbs sampling. Version 1.4.1*, Medical Research council Biostatistics Unit, Cambridge University, Cambridge, UK, 2004.

[21] R. Liu and R. Soetjipto, "Analysis of three Bayesian network inference algorithms: variable elimination, likelihood weighting, and Gibbs sampling," Tech. Rep., University of California, Berkeley, Calif, USA, 2004.

[22] T. Minka, *A family of algorithms for approximate Bayesian inference [Ph.D. thesis]*, Massachusetts Institute of Technology, 2001.

[23] J. Pearl, "Evidential reasoning using stochastic simulation of causal models," *Artificial Intelligence*, vol. 32, no. 2, pp. 245–257, 1987.

[24] T. Hrycej, "Gibbs sampling in Bayesian networks," *Artificial Intelligence*, vol. 46, no. 3, pp. 351–363, 1990.

[25] KDD CUP 2012, "Track 2: predict the click-through rate of ads given the query and user information," 2012, http://www.kddcup2012.org/c/kddcup2012-track2.

[26] Netica Application, "Norsys Software Corp," 2014, https://norsys.com/netica.html.