Learning Click Models via Probit Bayesian Inference

Yuchen Zhang[1,2]*, Dong Wang[1,2], Gang Wang[2], Weizhu Chen[2]
Zhihua Zhang[3], Botao Hu[1,2], Li Zhang[4]

[1]Institute for Theoretical Computer Science, Tsinghua University,
Beijing, China, 100084. {zhangyuc, wd890415, botao.a.hu}@gmail.com
[2]Microsoft Research Asia, Beijing, China, 100080.
{v-yuczha, v-dongmw, gawa, wzchen, v-boh}@microsoft.com
[3]College of Computer Science and Technology, Zhejiang University,
Hangzhou, Zhejiang, China, 310027. zhzhang@cs.zju.edu.cn
[4]School of Software, Tsinghua University, Beijing, China, 100084. lizhang@tsinghua.edu.cn

ABSTRACT

Recent advances in click models have positioned them as an effective approach to the improvement of interpreting click data, and some typical works include UBM, DBN, CCM, etc. After formulating the knowledge of user search behavior into a set of model assumptions, each click model developed an inference method to estimate its parameters. The inference method plays a critical role in terms of accuracy in interpreting clicks, and we observe that different inference methods for a click model can lead to significant accuracy differences. In this paper, we propose a novel Bayesian inference approach for click models. This approach regards click model under a unified framework, which has the following characteristics and advantages:

1. This approach can be widely applied to existing click models, and we demonstrate how to infer DBN, CCM and UBM through it. This novel inference method is based on the Bayesian framework which is more flexible in characterizing the uncertainty in clicks and brings higher generalization abilities. As a result, it not only excels in the inference methods originally developed in click models, but also provides a valid comparison among different models;

2. In contrast to the previous click models, which are exclusively designed for the position-bias, this approach is capable of capturing more sophisticated information such as BM25 and PageRank score into click models. This makes these models interpret click-through data more accurately. Experimental results illustrate that the click models integrated with more information can achieve significantly better performance on click perplexity and search ranking;

3. Because of the incremental nature of the Bayesian learning, this approach is scalable to process large scale and constantly growing log data.

*This work was done when the first author was visiting Microsoft Research Asia.

CIKM'10, October 26–29, 2010, Toronto, Ontario, Canada.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]:

General Terms

Algorithms, Experimentation, Performance

Keywords

Click Log Analysis, Click Model, Probit Bayesian Inference

1. INTRODUCTION

In a commercial search engine, terabytes of click-through logs are generated every day at very low cost. These click-through logs encode valuable user preferences with regard to search results and reveal the latest tendency of user click behaviors. Naturally, many studies have attempted to discover user preferences from click-through logs in order to improve web search ranking. Indeed, after the pioneering work of Joachims et al [11], which uses preferences automatically generated from click-through logs to train a ranking function, many interesting works have been proposed to estimate document relevance from user clicks [1, 2, 5, 14].

It has been noticed in existing works that one major difficulty in estimating relevance from click data comes from a so-called position bias: a document appearing in a higher position is more likely to attract user clicks even though it is not as relevant as documents in lower positions. Richardson et al [15] proposed to increase the relevance of documents in lower positions by a multiplicative factor. This idea was later formalized as the examination hypothesis [7] and adopted in the position model [6]. The examination hypothesis assumes the user will click a search result only after examining the search snippet. Craswell et al [7] extended the examination hypothesis and proposed the cascade model by assuming that the user will scan search results from top to bottom. Dupret and Piwowarski [8] introduced the positional distance into their UBM model. Recently, Guo et al [9] proposed the CCM model and Chappell et al [6] proposed the DBN model, both of which generalize the cascade model by assuming that the probability of examining the current document is related to the relevance of the document in the previous position.

Click models, such as UBM, DBN and CCM, have been demonstrated to be much more successful than the sim-

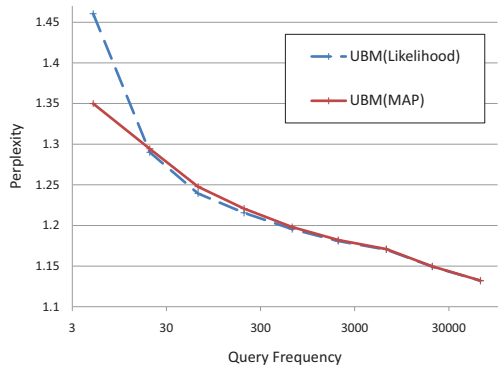


Figure 1: The perplexity score on different query frequencies achieved by the UBM model with maximum log-likelihood and maximum posteriori methods, respectively. Lower perplexity score indicates better prediction performance.

ple counting approach in interpreting click data. Each of these models introduced a set of assumptions integrating the knowledge of user browsing and click behaviors and developed an inference method. The inference methods in existing click models are often different with each other. For example, UBM used the EM algorithm to optimize the parameter through maximizing the likelihood function. DBN also used the EM algorithm, however, it tried to maximize the posterior function (MAP). CCM was a Bayesian approach and it approximated the posterior distribution through a multinomial distribution. We observe that the inference method plays an important role in terms of accuracy in click prediction. As illustrated in Figure 1, the click perplexity of UBM can be significantly improved for low frequent queries after switching from the maximum likelihood estimation to the maximum posterior estimation method. Thus, the difference of the inference methods in click models makes the comparison difficult. We cannot identify that the performance difference between two click models is due to either the model assumption or the inference method. Moreover, we find there is large space to develop a new inference approach for improving the existing model accuracy.

In this paper, we propose a novel inference approach which can be widely applied to existing click models. The new approach is based on the Bayesian framework. It replaces each probability variable in click models with a new variable following the Gaussian distribution through a probit link function, such that both the prior and the posterior distribution of the Bayesian learning can be approximated by Gaussians. We show that this inference approach is computationally tractable for all click models in which the likelihood functions are in the multinomial form. This requirement is general enough to be fitted into most of the existing click models.

We call the new proposed inference approach the Probit Bayesian Inference (PBI). Accordingly, the PBI approach can provide valid evaluation to compare different click models. We apply PBI to three state-of-the-art click models, such as UBM, DBN and CCM, and the experiments show that the new approach consistently achieves better performance than the original inference algorithm of these models.

Another challenge with previous click models is that they are designed for position-bias exclusively. However, we observe that a click may be affected by other factors besides the position. For example, a user click is affected by the relevance between a query and a snippet, which can be measured by the BM25 score. The PBI approach is capable of capturing more sophisticated information into a click model to interpret user clicks accurately. In this paper, we include seven measures such as BM25 and PageRank scores into the previous click models through PBI, and the experimental results demonstrate that the integration of these measures yields significant improvement in perplexity and relevance. Furthermore, PBI is an incremental approach, thus it is natural to handle very large-scale data set.

The paper is organized as follows: Section 2 briefly introduces previous works on click models including their specifications and hypothesis. In Section 3, the PBI approach will be presented in detail. In section 4, we give three examples on how the PBI approach is applied to the UBM, CCM and DBN click models. In section 5, we demonstrate how to integrate additional measures into the click models. Section 6 reports the experimental results and the conclusion follows.

2. PRELIMINARIES

The user starts a search session by submitting a *query* to the search engine, the search engine returning the user some ranked documents as search results. The user then browses the returned documents and clicks some of them. We use a binary random variable C_j to represent the click events of the document at position j. $C_j = 1$ indicates the user clicks the document at the jth position, while $C_j = 0$ indicates the user does not click this document. We assume that all queries and documents are indexed, so that we can use q_i and d_j to represent the the query and the document with the index i and j, respectively. Suppose that q_i is the query for the current session, the index of the document at the position j is represented by a mapping function $\phi(j)$.

2.1 Examination and Cascade Hypotheses

The examination hypothesis and the cascade hypothesis are proposed in order to simulate user browsing habits. Many existing click models are dependent on these two hypotheses. When a document is examined it means that the user has checked this document in the search results, and this event is denoted by a binary random variable E_j, in which j indicates the position of the document. $E_j = 1$ means that the document at position j is examined and $E_j = 0$ otherwise. The examination hypothesis assumes that a displayed document is clicked if and only if this document is both examined and perceived as relevant:

$$P(C_j = 1|E_j = 0) = 0$$
$$P(C_j = 1|E_j = 1) = R_{i\phi(j)}$$

where $R_{i\phi(j)}$ measures the degree of relevance between q_i and $d_{\phi(j)}$. The cascade hypothesis assumes that the user scans linear to the search results, thus, a document is examined only if all the above documents are examined. The first document is always examined:

$$P(E_{j+1} = 1|E_j = 0) = 0$$
$$P(E_1 = 1) = 1$$

2.2 CCM click model

The CCM model[9] assumes that user starts the examination of the search results from the top ranked document.

At each position j, the user can choose to click or skip the document $d_{\phi(j)}$ according to the perceived relevance. Either way, the user can choose to continue the examination or abandon the current query session. The probability of continuing to examine $d_{\phi(j+1)}$ depends on his action at the current position j:

$$
\begin{aligned}
&P(E_1 = 1) = 1 \\
&P(C_j = 1|E_j = 0) = 0 \\
&P(C_j = 1|E_j = 1, R_{i\phi(j)}) = R_{i\phi(j)} \\
&P(E_{j+1} = 1|E_j = 0) = 0 \\
&P(E_{j+1} = 1|E_j = 1, C_j = 0, \alpha_1) = \alpha_1 \\
&P(E_{j+1} = 1|E_j = 1, C_j = 1, \alpha_2, \alpha_3, R_{i\phi(j)}) \\
&= \alpha_2(1 - R_{i\phi(j)}) + \alpha_3 R_{i\phi(j)}
\end{aligned}
$$

2.3 DBN click model

The DBN model is designed based on the fact that a click does not necessarily indicate that the user is satisfied with this document. Thus, the DBN model[6] distinguishes the document relevance as the perceived relevance and the real relevance, where whether the user clicks a document depends on its perceived relevance while whether the user is satisfied with this document and examines the next document depends on the real relevance. Thus, the DBN click model is characterized as:

$$
\begin{aligned}
&P(E_1 = 1) = 1 \\
&P(C_j = 1|E_j = 0) = 0 \\
&P(C_j = 1|E_j = 1, a_{i\phi(j)}) = a_{i\phi(j)} \\
&P(S_j = 1|C_j = 0) = 0 \\
&P(S_j = 1|C_j = 1, s_{i\phi(j)}) = s_{i\phi(j)} \\
&P(E_{j+1} = 1|E_j = 0) = 0 \\
&P(E_{j+1} = 1|S_j = 1) = 0 \\
&P(E_{j+1} = 1|E_j = 1, S_j = 0, \gamma) = \gamma
\end{aligned}
$$

where S_j is a binary variable indicating whether the user is satisfied with the document $d_{\phi(j)}$ at position j, and the parameters $a_{i\phi(j)}$ and $s_{i\phi(j)}$ measure the perceived relevance and real relevance between q_i and $d_{\phi(j)}$, respectively.

2.4 UBM click model

Different from CCM and DBN, the UBM model[8] does not adopt the cascade hypothesis. Instead, it introduces a series of global parameters γ_{rd} to measure the probability that the user examines the document $d_{i\phi(r)}$ at position r after his last click at position $r - d$:

$$
\begin{aligned}
&P(E_r = 1|C_{1:r-1} = 0, \gamma_{rr}) = \gamma_{rr} \\
&P(E_r = 1|C_{r-d} = 1, C_{r-d+1:r-1} = 0, \gamma_{rd}) = \gamma_{rd} \\
&P(C_j = 1|E_j = 0) = 0 \\
&P(C_j = 1|E_j = 1, a_{i\phi(j)}) = a_{i\phi(j)}
\end{aligned}
$$

where $a_{i\phi(j)}$ measures perceived relevance. The term $C_{i:j} = 0$ is the abbreviation for $C_i = C_{i+1} = \cdots = C_j = 0$.

3. PROBIT BAYESIAN INFERENCE

Typically, a click model $\mathcal{M}$ is parameterized by a set of unknown variables $\theta_1, \cdots, \theta_n$. The performance of $\mathcal{M}$ relies on the values of these parameters. However, since it usually assumes $\theta_i \in (0, 1)$ for $i = 1, \ldots, n$, this would limit the flexibility of inference algorithms. In this section, we propose a new framework for handling this limitation. The key idea is to associate each θ_i with a Gaussian auxiliary variable $x_i \in (-\infty, +\infty)$ via a so-called probit link. This approach also facilitates the incorporation of more sophisticated information such as PageRank and BM25 into the click model, thus it significantly enhances the generalization of the resulting model (details in Section 5).

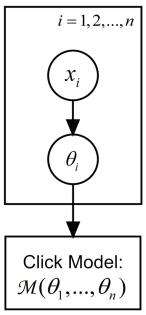


Figure 2: Graphical representation of the probit Bayesian inference approach, in which $\theta_1, \cdots, \theta_n$ are probability parameters of the click model $\mathcal{M}$. Each parameter θ_i is connected to a Gaussian distributed variable x_i via the probit link $\theta_i = \Phi(x_i)$. Here, $\Phi(x) = \int_{-\infty}^{x} \mathcal{N}(t; 0, 1)\mathrm{d}t$ is the normal cumulative distribution function.

3.1 Framework

We are given a click model $\mathcal{M}$ on a series of query sessions. When a session is loaded in, we assume that this session contains M impressions, in which $C_j \in \{0, 1\}$ indicates whether the j-th impression has been clicked. Let $C_{j:k}$ denote the vector $(C_j, \ldots, C_k)$, we define the likelihood function $P(C_{1:M}|\theta_1, \cdots, \theta_n)$ as

$$P(C_{1:M}|\theta_1, \ldots, \theta_n) = f(\theta_1, \ldots, \theta_n). \quad (1)$$

In this paper, we assume that f is a polynomial function of $\theta_1, \cdots, \theta_n$. This assumption is satisfied in most existing click models, such as the Cascade model, UBM, DCM, CCM and DBN.

In our framework, we introduce the auxiliary variables x_i for each θ_i. The connection between θ_i and x_i is further defined by

$$\theta_i = \Phi(x_i),\ i = 1, \ldots, n$$

where $\Phi(x) = \int_{-\infty}^{x} \mathcal{N}(t; 0, 1)\mathrm{d}t$, the cumulative distribution function of the standard normal distribution is referred to as the probit link [3, 4]. Thus, we rewrite the likelihood function in (1) as

$$P(C_{1:M}|x_1, \cdots, x_n) = f\left(\Phi(x_1), \ldots, \Phi(x_n)\right). \quad (2)$$

Given a session, the *order* of $\Phi(x_i)$ is defined as its highest-order power in (2). Moreover, x_i is called an *active* variable if the order of $\Phi(x_i)$ is non-zero. Furthermore, we assume that x_i independently comes from the Gaussian distribution $\mathcal{N}(x_i; \mu_i, \sigma_i^2)$. Figure 2 illustrates the framework.

In this framework, the motivation of using the probit link instead of using other links (the logistic link, for example), is mainly due to the desirable computational property provided by the probit link. More specifically, it is because that there are several non-trivial integration steps involved

in the inference, whose computational tractability and efficiency relies on the close relationship between the probit link and the Gaussian distribution. For example, in order to compute the marginal distribution of a specific variable from the joint distribution, we have to integrate out all other variables; furthermore, when we try to use the variational method to approximate a density function to be the Gaussian density, we will encounter a similar integration problem. In these cases, if the function to be integrated does not hold an appropriate property, the integration may become very inefficient or even intractable. Recall that the inference algorithm in this paper is designed to process terabytes amounts of data, so we have to make sure that the computation can be carried out very fast, as well as with sufficiently high precision. In the Appendix, we introduce an efficient integration algorithm to handle these problems. As suggested above, this algorithm requires that the probit link is employed instead of other links.

In addition, since we allow the form of the likelihood function f to be arbitrary polynomial, the computational tractability is also a big reason to encourage us to propose a Bayesian framework for click model inference, instead of other methods such as logistic regression.

3.2 Inference

We now develop an inference algorithm for the framework. That is, we want to estimate the parameters (μ_i, σ_i^2) from the posterior distributions $p(x_i|C_{1:M}, \mu_i, \sigma_i^2)$. For this purpose, we use an online learning scheme referred to as Gaussian density filtering [13]. This scheme first approximates $p(x_i|C_{1:M}, \mu_i, \sigma_i^2)$ as a Gaussian distribution and then updates the estimates of (μ_i, σ_i^2) based on this Gaussian. For each query session, this updating procedure is executed once.

For a specific active variable x_i, the posterior distribution of x_i under the click observation $C_{1:M}$ is

$$p(x_i|C_{1:M}, \mu_i, \sigma_i^2) \propto p(x_i|\mu_i, \sigma_i^2) P(C_{1:M}|x_i), \tag{3}$$

where $p(x_j|\mu_i, \sigma_j^2) = \mathcal{N}(x_j; \mu_j, \sigma_j^2)$ is the prior distribution of x_j. The marginal likelihood function of x_i is then obtained by integrating out all x_j's but x_i from the function f, namely

$$\begin{aligned} &P(C_{1:M}|x_i) \\ &= \int_{\mathbb{R}^{n-1}} f\left(\Phi(x_1), \cdots, \Phi(x_n)\right) \prod_{j \neq i} (p(x_j|\mu_j, \sigma_j^2) \mathrm{d}x_j). \end{aligned} \tag{4}$$

Suppose that the function (2) is expanded as the form

$$f(\Phi(x_1), \cdots, \Phi(x_n)) = \sum_{t=1}^{T} \left(\alpha_t \prod_{j=1}^{n} \Phi^{k_{tj}}(x_j) \right),$$

where k_{tj} is the power of $\Phi(x_j)$ in the t-th term of the expansion. Accordingly, we write the integral in (4) as

$$P(C_{1:M}|x_i) = \sum_{t=1}^{T} \Big(\alpha_t \Phi^{k_{ti}}(x_i) \prod_{j \neq i} c_{jk_{tj}} \Big), \tag{5}$$

where

$$c_{ik} = \int_{-\infty}^{+\infty} \Phi^k(x_i) p(x_i|\mu_i, \sigma_i^2) \mathrm{d}x_i, \tag{6}$$

for all active variables x_i and all powers $0 \leq k \leq K_i$. Here K_i is the order of $\Phi(x_i)$. The details of computing c_{ik} are given in the Appendix. We now rearrange (5) into the standard form

$$P(C_{1:M}|x_i) = a_{K_i} \Phi^{K_i}(x_i) + \cdots + a_1 \Phi(x_i) + a_0. \tag{7}$$

We can see from (3) and (7) that $p(x_i|C_{1:M}, \mu_i, \sigma_i^2)$ is no longer Gaussian. This makes the inference inefficient. The idea behind the Gaussian density filtering method is to approximate $p(x_i|C_{1:M}, \mu_i, \sigma_i^2)$ by a Gaussian distribution $\mathcal{N}(x_i; \hat{\mu}_i, \hat{\sigma}_i^2)$ (denoted by $q(x_i|\hat{\mu}_i, \hat{\sigma}_i^2)$) and treat it as the prior distribution for the next session.

In order to find $q(x_i|\hat{\mu}_i, \hat{\sigma}_i^2)$, we attempt to minimize the Kullback-Leibler (KL) divergence between $p(x_i|C_{1:M}, \mu_i, \sigma_i^2)$ and $q(x_i|\hat{\mu}_i, \hat{\sigma}_i^2)$. This minimization involves computing the first three-order moments of $p(x_i|C_{1:M}, \mu_i, \sigma_i^2)$, which can be reduced to the evaluation of the integrals in (6) and

$$u_{ik} = \int_{-\infty}^{+\infty} x \Phi^k(x_i) p(x_i|\mu_i, \sigma_i^2) \mathrm{d}x_i, \tag{8}$$

$$v_{ik} = \int_{-\infty}^{+\infty} x^2 \Phi^k(x_i) p(x_i|\mu_i, \sigma_i^2) \mathrm{d}x_i. \tag{9}$$

It is worth pointing out that it is not trivial to compute the integrals in (6), (8) and (9). In the Appendix, we devise an efficient approach to the computation of (6), (8) and (9) by using Expectation Propagation [13] and the iterative message passing on factor graphs[12].

It then follows from (7) that $\hat{\mu}_i$ and $\hat{\sigma}_i^2$ is given by

$$\hat{\mu}_i = \frac{\sum_{k=0}^{K_i} a_k u_{ik}}{\sum_{k=0}^{K_i} a_k c_{ik}}, \tag{10}$$

$$\hat{\sigma}_i^2 = \frac{\sum_{k=0}^{K_i} a_k v_{ik}}{\sum_{k=0}^{K_i} a_k c_{ik}} - \hat{\mu}_i^2. \tag{11}$$

Thus, we complete the updating procedure (μ_i, σ_i^2) for x_i. Similar updates should be performed on all active variables before the next session is loaded into the model.

The inference algorithm is summarized in Algorithm 1. It is an incremental updating algorithm as query sessions are sequentially loaded into the click model.

Algorithm 1 The Inference Algorithm

```
for each query session do
    Derive the likelihood function (2).
    For each active variable x_i ∈ {x_1, ··· , x_n}, compute
    c_ik, u_ik and v_ik through 0 ≤ k ≤ K_i, where K_i is the
    order of Φ(x_i).
    for each active variable x_i ∈ {x_1, ··· , x_n} do
        Evaluate P(C_1:M|x_i) according to (5).
        Approximate p(x_i|C_1:M, μ_i, σ_i^2) by Gaussian distribution with mean μ̂_i and variance σ̂_i^2 given in (10)
        and (11).
        Update the parameters μ_i and σ_i^2 by setting μ_i ←
        μ̂_i and σ_i^2 ← σ̂_i^2.
    end for
end for
```

3.3 Prediction

After the (μ_i, σ_i^2) have been obtained, the value of each parameter θ_i is estimated by the expectation of $\Phi(x_i)$ with

respect to $p(\mathbf{x}_i|\mu_i,\sigma_i^2)$; that is, θ_i is given by

$$\theta_i = \int_{-\infty}^{+\infty} \Phi(x_i)\mathcal{N}(x_i;\mu_i,\sigma_i^2)\mathrm{d}x_i = \Phi\left(\frac{\mu_i}{\sqrt{\sigma_i^2+1}}\right).$$

With the estimated θ_i, the click model $\mathcal{M}(\theta_1,\ldots,\theta_n)$ can make predictions following its own predictive algorithm.

4. CASE STUDIES

We demonstrate how specific click models are inferred by using the probit Bayesian inference. According to the PBI framework defined in Section 3.1, it is sufficient to show how the likelihood function (2) is derived from each click model, so that the general inference algorithm in Section 3.2 can be immediately adopted. In this section, queries and documents are indicated by q_i and d_j, where i or j indicates the index of a specific query or document. We assume that q_i is the query for the current session, and $\phi(j)$ indicates the index of the document at the j position. There are M documents in the current session. Moreover, since the probit Bayesian inference introduces a hierarchical-style framework (see Figure 2), we call the click model $\mathcal{M}$ inferred under PBI the *hierarchical* $\mathcal{M}$, and abbreviated as H-$\mathcal{M}$.

4.1 Hierarchical UBM (H-UBM)

For UBM, the relevance parameters a_{kl} and the global parameters γ_{rd} are defined via probit links, i.e., $a_{kl}=\Phi(u_{kl})$ and $\gamma_{rd}=\Phi(\omega_{rd})$. Therefore the likelihood function is

$$\prod_{j=1}^{M}(\Phi(u_{i\phi(j)})\Phi(\omega_{jd_j}))^{C_j}(1-\Phi(u_{i\phi(j)})\Phi(\omega_{jd_j}))^{1-C_j} \quad (12)$$

where d_j represents the distance between the position j and the last clicked position before j. If there is no click before the jth position, we set $d_j=j$. This likelihood function is obviously polynomial. For active u or ω, the orders of $\Phi(u)$ and $\Phi(\omega)$ are always 1.

4.2 Hierarchical CCM (H-CCM)

For CCM, the relevance parameters R_{kl} and the global parameters $\alpha_1,\alpha_2,\alpha_3$ are defined as $R_{kl}=\Phi(r_{kl})$ and $\alpha_k=\Phi(\omega_k)$. In the current session, there are $M+3$ active parameters: $r_{i\phi(j)}$ $(j=1,\cdots,M)$ and ω_l $(l=1,2,3)$. The likelihood function is given by

$$P(C_{1:M}|r_{i\phi(1)},\cdots,r_{i\phi(M)},\omega_1,\omega_2,\omega_3), \quad (13)$$

which is a polynomial function of $\Phi(r)$'s and $\Phi(\omega)$'s. Furthermore, the order of $\Phi(r_{i\phi(j)})$ is at most 2, and the order of $\Phi(\omega_j)$ is at most M. If we use $p_j^{(e)}(\mathbf{r},\omega)$ to represent the function $P(C_{1:j},E_{j+1}=e|r_{i\phi(1)},\ldots,r_{i\phi(M)},\omega_1,\omega_2,\omega_3)$, then (13) can be derived via recursive formulas based on the specification of CCM:

$$\begin{aligned}
&p_0^{(0)}(\mathbf{r},\omega)=0;\quad p_0^{(1)}(\mathbf{r},\omega)=1;\\
&p_j^{(0)}(\mathbf{r},\omega)=(1-C_j)p_{j-1}^{(0)}(\mathbf{r},\omega)+p_{j-1}^{(1)}(\mathbf{r},\omega)\\
&\qquad\times\begin{cases}(1-\Phi(\omega_1))(1-\Phi(r_{i\phi(j)})) & C_j=0,\\ (1-\Phi(\omega_2))\Phi(r_{i\phi(j)}) & \\ \quad+(\Phi(\omega_2)-\Phi(\omega_3))\Phi^2(r_{i\phi(j)}) & C_j=1;\end{cases}\\
&p_j^{(1)}(\mathbf{r},\omega)=p_{j-1}^{(1)}(\mathbf{r},\omega)\\
&\qquad\times\begin{cases}\Phi(\omega_1)(1-\Phi(r_{i\phi(j)})) & C_j=0,\\ \Phi(\omega_2)\Phi(r_{i\phi(j)}) & \\ \quad+(\Phi(\omega_3)-\Phi(\omega_2))\Phi^2(r_{i\phi(j)}) & C_j=1.\end{cases}
\end{aligned}$$

It is straightforward to see that $p_M^{(0)}(\mathbf{r},\omega)+p_M^{(1)}(\mathbf{r},\omega)$ is exactly equal to (13).

4.3 Hierarchical DBN (H-DBN)

For DBN, the perceived relevance parameters a_{kl} are defined as $a_{kl}=\Phi(u_{kl})$, while the actual relevance parameters s_{kl} are defined as $s_{kl}=\Phi(v_{kl})$. The likelihood function

$$P(C_{1:M}|u_{i\phi(1)},\cdots,u_{i\phi(M)},v_{i\phi(1)},\cdots,v_{i\phi(M)}) \quad (14)$$

is also derived by recursion. If we use $p_j^{(e)}(\mathbf{u},\mathbf{v})$ to represent the function

$$P(C_{1:j},E_{j+1}=e|u_{i\phi(1)},\cdots,u_{i\phi(M)},v_{i\phi(1)},\cdots,v_{i\phi(M)}),$$

then the following recursions hold:

$$\begin{aligned}
&p_0^{(0)}(\mathbf{u},\mathbf{v})=0;\quad p_0^{(1)}(\mathbf{u},\mathbf{v})=1;\\
&p_j^{(0)}(\mathbf{u},\mathbf{v})=(1-C_j)p_{j-1}^{(0)}(\mathbf{u},\mathbf{v})+p_{j-1}^{(1)}(\mathbf{u},\mathbf{v})\\
&\qquad\times\begin{cases}(1-\gamma)(1-\Phi(u_{i\phi(j)})) & C_j=0,\\ \Phi(u_{i\phi(j)})(1-\gamma+\gamma\Phi(v_{i\phi(j)})) & C_j=1;\end{cases}\\
&p_j^{(1)}(\mathbf{u},\mathbf{v})=p_{j-1}^{(1)}(\mathbf{u},\mathbf{v})\\
&\qquad\times\begin{cases}\gamma(1-\Phi(u_{i\phi(j)})) & C_j=0,\\ \gamma\Phi(u_{i\phi(j)})(1-\Phi(v_{i\phi(j)})) & C_j=1.\end{cases}
\end{aligned}$$

It is straightforward to see that $p_M^{(0)}(\mathbf{u},\mathbf{v})+p_M^{(1)}(\mathbf{u},\mathbf{v})$ is exactly equal to (14). In DBN, the order of $\Phi(u)$ or $\Phi(v)$ is always 1, if it is positive.

5. DEEP PROBIT BAYESIAN INFERENCE

In the PBI approach of Section 3, we assume that the auxiliary variables x_i follow the Gaussian distributions. In order to capture more sophisticated information into the click model, we develop a Deep Probit Bayesian Inference approach (Deep-PBI). That is, we further extend the framework described in Figure 2 to the framework shown in Figure 3. In this new framework, each auxiliary variable x_i is interpreted as a linear combination of several factors. Each of these factors follows the Gaussian distribution. In this paper, we define two kinds of factors: the historical mirror and the weighted combination of additional measures.

The historical mirror is written as h_i. Its distribution corresponds to the posterior distribution of x_i computed in its last training epoch. Note that h_i should be initialized by a default Gaussian before the first training epoch of x_i.

The weighted combination of additional measures is written as the inner product $\mathbf{y}^T\mathbf{w}$ of two vectors $\mathbf{y}$ and $\mathbf{w}$. Here

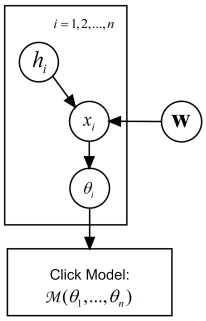


Figure 3: Graphical representation of the deep probit Bayesian inference approach. This framework is capable of integrating additional measures into click models.

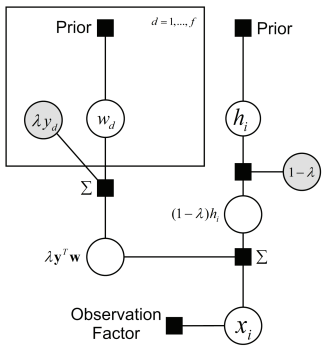


Figure 4: Factor graph for updating h_i and $\mathbf{w}$. The large rectangles or *plates* indicate parts of the graph which are repeated with repetition indexed by the variable in the corner of the plate. The factors labeled Σ are sum factors of the form $I[z = x + y]$.

$\mathbf{y}$ represents the values of measures extracted from the current session, which may contain the background description associated with x_i, such as the BM25 score and the PageRank score. $\mathbf{w}$ is a global vector, regulating the weight of each measure. Each element of $\mathbf{w}$ is a Gaussian variable, which is updated throughout the training process. Thus, x_i is defined by

$$x_i = (1 - \lambda(h_i))h_i + \lambda(h_i)\mathbf{y}^T\mathbf{w},$$

where $\lambda(h_i)$ is a function of h_i and regulates the proportion of contribution of additional measures. Generally speaking, $\lambda(h_i)$ should be a monotonically decreasing function of $\sigma^2(h_i)$, which means that the value of additional measures gives a major contribution to x_i only if we have little confidence on the precision of h_i.

The factor graph in Figure 4 is constructed according to the joint distribution of x_i, h_i and $\mathbf{w}$. Based on this graph, the prior of x_i can be evaluated from the priors of h_i and $\mathbf{w}$ following the sum-product algorithm [12], computed in a forward manner. Once we obtain the prior of x_i, we can apply the inference algorithm in Section 3 to do inference and prediction.

In the deep probit Bayesian inference, the posterior of x_i is approximated by a Gaussian distribution $\mathcal{N}(x_i; \hat{\mu}_i, \hat{\sigma}_i^2)$. Moreover, this Gaussian posterior is used for setting the observation factor of the factor graph, instead of updating x_i directly. The observation factor is given by

$$m_{o \to x_i}(x_i) = \frac{\mathcal{N}(x_i; \hat{\mu}_i, \hat{\sigma}_i^2)}{p(x_i)},$$

where $p(x_i)$ represents the prior of x_i. Then, the sum-product algorithm is applied once again to compute the posterior marginal distribution for each element of $\mathbf{w}$, in a backward manner. Since all factors in the factor graph are Gaussian, the computation is very efficient. The final step is to update the mean and variance of each weight according to its posterior. On the other hand, the mean and variance of h_i is updated by $\hat{\mu}_i$ and $\hat{\sigma}_i^2$.

The deep probit Bayesian framework proposed in this section leads to the deep hierarchical version of UBM, CCM and DBN, which are denoted by H^2-UBM, H^2-CCM and H^2-DBN.

Table 1: The summary of the data set in the experiment

Query Frequency	# Query	# Train Session	# Test Session
1 - 10	2,813	7,775	5,978
10 - 30	2,671	26,227	24,659
30 - 100	2,512	78,108	76,638
100 - 300	2,214	211,765	210,434
300 - 1,000	2,173	665,657	664,372
1,000 - 3,000	749	731,545	731,005
3,000 - 10,000	432	1,303,026	1,302,551
10,000 - 30,000	171	1,579,869	1,579,439
>30,000	34	1,500,306	1,500,191
above all	13,679	6,104,278	6,095,267

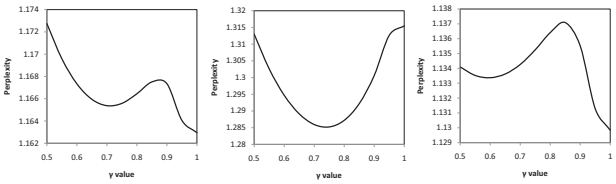
6. EXPERIMENTS

In the experiment, we include three state-of-the-art models: UBM, CCM and DBN, into the evaluation. The experimental results are compared between the original inference algorithm of click models, the probit Bayesian inference, and the deep probit Bayesian inference (with additional information integrated). There are three parts of the evaluation considered in this paper. In this first part, the click perplexity is evaluated to demonstrate the performance improvement after the previous click models are inferred using the new approach. Moreover, we give a comparison among different click models inferred by the same version of PBI. In the second part, we use NDCG to evaluate the relevance obtained from click models. Finally, the efficiency of the PBI and Deep-PBI approach is discussed. All of the experiments were carried out on a 64-bit server with 32GB RAM and sixteen 2.53GHz processors.

6.1 Experimental Setup

The click logs used to train the click models are collected from a large commercial search engine which comprises 13,679 queries and 12.2 million sessions. The dataset is divided randomly and evenly into a training set with 6,104,278 sessions and a testing set with 6,095,267 sessions. When generating the training and testing data, we restrict all queries in the testing set to have at least one session included in the training set. Table 1 reports the number of queries with respect to the query frequency.

In the experiments, the original inference algorithm of UBM, CCM and DBN exactly follows their conventions in [8, 9, 6]. The original DBN inference requires a input of the



(a) All Query (b) Low Freq. Query (c) High Freq. Query

Figure 5: Perplexity of the DBN model as a function of γ. The leftmost figure is on all queries; the middle figure is on the queries with frequencies lower than 10; the rightmost curve is on the queries with frequencies higher than $30,000$.

parameter γ. To determine this input, the click perplexities (see definition in Section 6.2) on the entire data set are evaluated using the original DBN inference algorithm with a series of distinct γ values. We found the optimal value of γ is 1, while $\gamma = 0.7$ is a local minimum, as shown in Figure 5(a). To understand this phenomenon, we plot the curve of the perplexity on the queries with the lowest and the highest frequency respectively for distinct γ, which are ploted in Figure 5(b) and Figure 5(c). The curves show that the optimal γ is not uniform on different frequencies. According to the figure, the optimal γ is close to 0.7 for low frequency queries, but this value is 1 for high frequency queries. Hence, in order to guarantee fairness in experiments, we evaluate the performance of DBN on both $\gamma = 0.7$ and $\gamma = 1$. For the original CCM inference algorithm, the global parameters $\alpha_1, \alpha_2, \alpha_3$ are trained following the method introduced in [9]. Since the original inference algorithm of CCM is not capable of handling highly frequent queries, the experiments for CCM are restricted on the queries with frequencies less than $3,000$, which is in accordance with the experimental setup in [9].

When click models are inferred using the PBI approach, besides all perceived relevance or actual relevance parameters, global parameters such as γ_{rd} of UBM and $\alpha_1, \alpha_2, \alpha_3$ of CCM are also trained under the PBI approach. The prior distributions of all variables are initialized to the standard normal distribution. In other words, we did not include any prior knowledge on the data set. The click models inferred by their original inference algorithms (named as UBM, CCM and DBN as usual), the click models inferred by PBI (named as H-UBM, H-CCM and H-DBN, as described in Section 3) and the click models inferred by Deep-PBI (named as H^2-UBM, H^2-CCM and H^2-DBN, as described in Section 5) are evaluated separately. When additional measures are integrated, the historical mirrors are initialized to follow the standard normal distribution. Moreover, we define

$$\lambda(h_i) = \begin{cases} 0.5 & \sigma^2(h_i) = 1 \\ 0 & \sigma^2(h_i) < 1 \end{cases}$$

In other words, the weights of measures are updated and take effect only when a variable x_i is activated at the first time. This strategy restricts the influence of additional measures only for initialization, and has been shown to be very effective in the experiment.

There are seven measures extracted in the experiment, which are summarized in the below table.

Table 2: The percentage on the click perplexity improvement achieved by Deep-PBI (with seven additional measures integrated).

Frequency	UBM	CCM	DBN ($\gamma = 0.7$)	DBN ($\gamma = 1$)
1-10	41.7%	9.39%	6.18%	14.4%
10-30	13.3%	7.69%	3.65%	9.59%
30-100	3.09%	6.09%	1.99%	4.66%
100-300	0.24%	5.03%	1.33%	1.97%
300-1,000	0.19%	4.16%	1.23%	0.98%
1,000-3,000	0.58%	3.99%	1.46%	0.62%

Name	Meaning
WordsFoundUrl	$\frac{\#\text{key words found in url}}{\#\text{words in query}}$.
WordsFoundTitle	$\frac{\#\text{key words found in document title}}{\#\text{words in query}}$.
WordsFoundBody	$\frac{\#\text{key words found in document body}}{\#\text{words in query}}$.
BM25Norm	The normalized BM25 Score.
PageRank	A measure on a document quality.
DomainRank	Another measure on a document quality.
CurrentPosition	Position of the document displayed at the current search result page.

In experiments, the vector $\mathbf{y}$ is the catenation of seven sparse binary vectors $\mathbf{y} = \mathbf{f_1 f_2} \cdots \mathbf{f_7}$. Since the number of possible values of each measure is finite, we can give all possible values an order and denote the j-th possible value of the i-th measure by v_{ij}. Then we set the j-th element of $\mathbf{f}_i$ to 1 if v_{ij} is the current value of the i-th measure, and set to 0 otherwise. All weight variables are initialized to the normal distribution $\mathcal{N}(0, \frac{1}{7})$, so that the inner product $\mathbf{y}^T\mathbf{w}$ in the initialization satisfies the standard normal distribution.

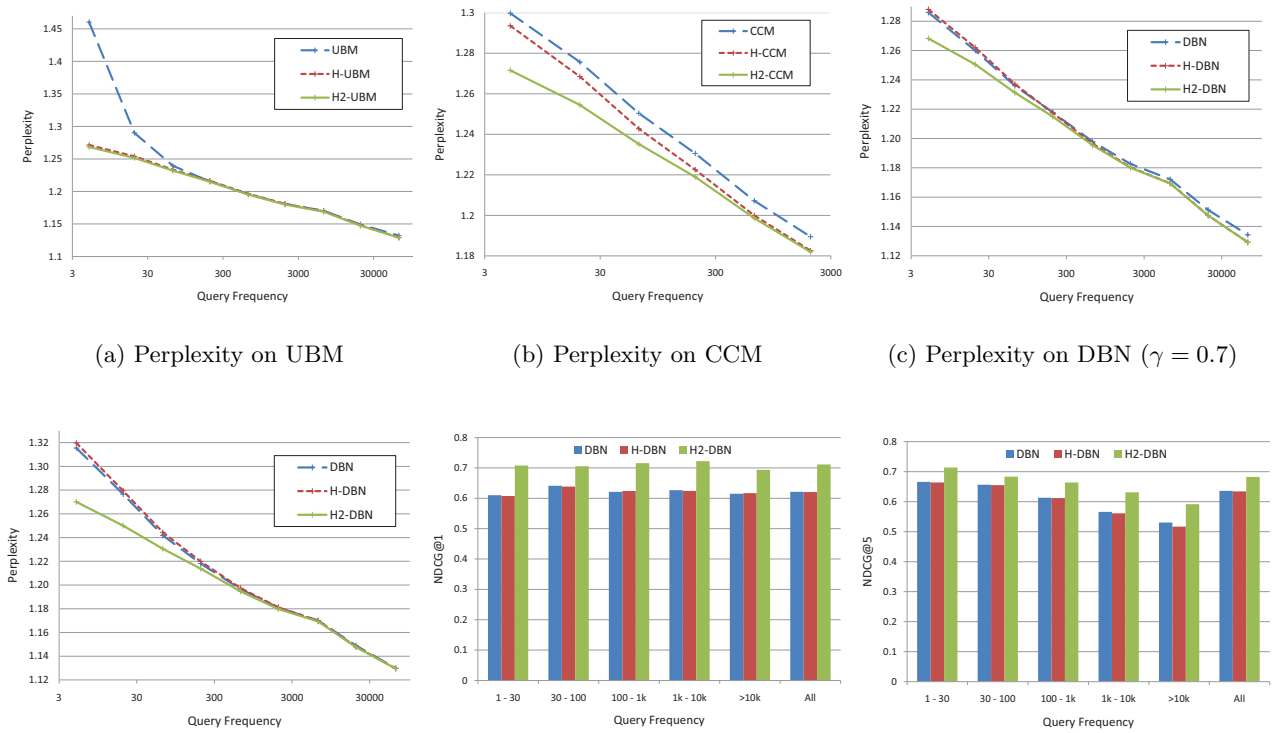
6.2 Click Perplexity

Click perplexity is widely used to measure model accuracy in click prediction. It has been used as the evaluation metric in [8, 9] for the UBM and CCM click models. It is computed for binary click events at each position of a query session independently. We assume that q_j^i is the probability of the click derived from the click model, i.e. $P(C_j^i = 1)$ at position j and C_j^i is a binary value indicating the click event at position j on the ith session. Thus, the click perplexity at position j is computed as follows:

$$p_j = 2^{-\frac{1}{N}\sum_{n=1}^{N}(C_j^n log_2 q_j^n + (1-C_j^n) log_2 (1-q_j^n))}$$

The perplexity of a data set is defined to be the average of all position perplexities. Thus, a smaller perplexity value indicates a better prediction. The improvement of perplexity value p_1 over p_2 is given by $\frac{p_2-p_1}{p_2-1} \times 100\%$.

The click perplexity on the testing set is reported in Figure 6 over different frequencies. It is observed that H^2-UBM, H^2-CCM and H^2-DBN significantly and consistently outperform the original version of UBM, CCM and DBN. The performance of H-UBM and H-CCM is also much better than the original UBM and CCM, while H-DBN is fairly comparable with the original DBN on both $\gamma = 0.7$ and $\gamma = 1$. The perplexity improvement achieved by PBI and Deep-PBI are relatively higher on low-frequency queries, which is summarized in Table 2. In the original UBM inference, the document that is never clicked in the training set will eventually have its relevance converging to zero. However, if this document is clicked in the testing set, the corresponding perplexity punishment will be considerable. This explains the poor performance of UBM on low frequency queries. As shown in Figure 6(a), the Bayesian inferred version, either H-UBM or



(a) Perplexity on UBM (b) Perplexity on CCM (c) Perplexity on DBN ($\gamma = 0.7$)

(d) Perplexity on DBN ($\gamma = 1$) (e) NDCG@1 on DBN ($\gamma = 1$) (f) NDCG@5 on DBN ($\gamma = 1$)

Figure 6: Perplexity and NDCG test on the test data for different query frequencies. In these graphs, H-UBM, H-CCM and H-DBN stand for the models inferred by PBI ; H^2-UBM, H^2-CCM and H^2-DBN stand for the models inferred by Deep-PBI (with seven additional measures integrated).

H^2-UBM, achieves a good perplexity score on both low and high frequency queries. Since the gap between the H-UBM curve and the H^2-UBM curve is small, we can conclude that the additional measures are not very helpful to UBM.

For CCM, the curves of H-CCM and H^2-CCM are significantly lower than the original CCM curve in both high and low frequency queries, as shown in Figure 6(b). Moreover, the integration of seven measures helps a lot on low frequency queries. However, as query frequency increases, the improvement provided by additional measures continuously fades and eventually disappears.

For DBN, we observed from Figure 6(c)(d) that the curve of H-DBN is fairly close to that of the original DBN. In other words, the Bayesian inference and the EM inference have similar performance on the DBN model. However, as additional measures are integrated, H^2-DBN performs much better than the original DBN on low frequency queries. When $\gamma = 0.7$, the original DBN gives a relatively precise prediction on low frequencies, but not precise enough on high frequency queries. Thus, it does not gather the best perplexity score on the entire dataset, as shown in Figure 5. On the other hand, we discovered that H-DBN with $\gamma = 0.7$ does not hold this defect, which indicates the Bayesian inference is less sensitive to inappropriate values of γ on high frequency queries.

One advantage of the PBI approach is that it provides a convincing comparison among click models, because after a click model is inferred by the new approach, none of the model assumptions are changed and the inference method

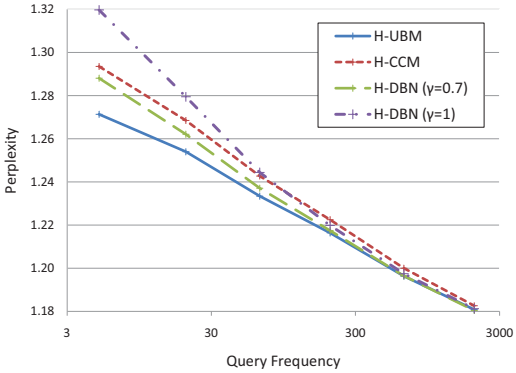


Figure 7: Performance comparison between H-UBM, H-CCM and H-DBN on perplexity metrics. The test is reported on queries with frequencies no more than 3,000.

becomes the same. In Figure 7, this comparison is reported on H-UBM, H-CCM and H-DBN. Since CCM is evaluated on queries with frequencies less than $3,000$ and other models perform very similarly on higher frequency queries, thus, in this experiment the comparison is reported on the range of frequencies that CCM can handle. It is observed that H-UBM gains the best perplexity score on all frequencies, followed by the $\gamma = 0.7$ version of H-DBN. This is because the PBI approach improves the performance of UBM on low frequencies, and improves the performance of DBN on high frequencies. The perplexity score achieved by H-CCM is

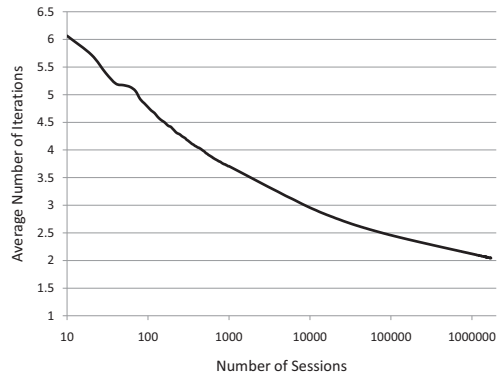


Figure 8: Average number of iterations when c_{ik}, u_{ik} and v_{ik} are computed following the iterative algorithm in the Appendix. This curve is plotted in the training process of H-CCM.

a bit worse than H-DBN($\gamma = 0.7$), which implies that the assumptions of CCM are not as reasonable as that of UBM and DBN on this dataset. The $\gamma = 1$ version of H-DBN gets the worst perplexity score on low frequencies, which implies that $\gamma = 1$ is not a reasonable setting for low frequency queries.

6.3 NDCG

The Normalized Discounted Cumulated Gain (NDCG)[10] is used as a metric for DBN[6] to measure the document relevance inferred for search ranking. In the NDCG evaluation, we compare DBN with H-DBN and H^2-DBN. The relevance obtained from UBM and CCM are not reported because UBM and CCM are designed to estimate the document perceived relevance rather than the actual relevance. Thus, both of them perform significantly worse than DBN on NDCG metrics.

Here, the γ parameter is fixed at 1 because $\gamma = 1$ leads to much better NDCG scores than $\gamma = 0.7$ (the $\gamma = 1$ version is 13.2% and 9.4% better than the $\gamma = 0.7$ version on NDCG@1 score and NDCG@5 score over all frequency queries). For each query, we rank the returned documents according to the actual relevance $a_{i\phi(j)} \cdot s_{i\phi(j)}$, then the NDCG scores are evaluated on queries and related documents whose relevance ratings exist in the HRS (Human Relevance System), where professional editors provide a five grade rating between a query and a document (4: perfect, 3: excellence, 2: good, 1: fair, 0: bad).

The NDCG scores are reported in Figure 6. Similar to the result in click perplexity, H-DBN achieves similar NDCG scores with the original DBN model. However, when additional measures are integrated in, the NDCG scores are substantially increased. The NDCG@1 scores of DBN, H-DBN and H^2-DBN are 0.6215, 0.6207 and 0.7113 over all frequencies; the corresponding NDCG@5 scores are 0.6361, 0.6342 and 0.6826. Compared with the performance from H^2-DBN and DBN, H^2-DBN is 14.6% better on NDCG@1 and 7.63% better on NDCG@5, both of which are considered to be very significant improvements.

6.4 Efficiency

The efficiency of Algorithm 1 is mainly determined by the efficiency of its third step. That is, the procedure in which c_{ik}, v_{ik} and v_{ik} are computed. These computations are archived by the evaluation algorithm presented in the Appendix. For UBM and DBN, the order of $\Phi(x)$ is always 0 or 1, hence the evaluation algorithm is never iterative, which guarantees the whole inference efficient. For CCM, the order of $\Phi(x)$ may equal to 2 when x represents the document relevance, or be even higher when x represents the global parameters. In these cases, the assignments of Step A and Step B in the Appendix may need to be iterated for several times before μ_B and σ_B converges. Obviously, the average number of iterations should be small enough to keep the whole inference efficient.

In the experiment, we require that the iterative assignment terminates when μ_B and σ_B varies no more than 10^{-8} between two consecutive iterations. The average number of iterations for computing c_{ik}, u_{ik} and v_{ik} is then evaluated in the training process of H-CCM and the result is reported in Figure 8. As we see, this number continuously decreases as an increasing number of sessions are processed. After the training finishes, we find that the average number of iterations is only 2.04. According to this result, and also considering the incremental nature of our approach, we can conclude that the efficiency of PBI and Deep-PBI is competitive compared with the original inference algorithm of click models.

7. CONCLUSION

In this paper, we have proposed a probit Bayesian inference approach for click models and have applied it to the DBN, CCM and UBM models. We have developed a framework that can be applied on most of the existing click models, and an efficient algorithm for training and prediction. The experiments have demonstrated that new approach achieves higher generalization ability than the original inference algorithm of previous click models. Moreover, this new approach has provided a unified inference method so that different click models have been compared with each other. Our proposed approach has been capable of integrating more sophisticated information into a click model through a deep hierarchical extension, and it has provided a significantly better performance on click perplexity and search ranking. In our future work, we will further explore its potential applications in click models.

Ackownledgement

This work was supported in part by the National Basic Research Program of China Grant Nos.2007CB807900, 2007CB 807901, the National Natural Science Foundation of China Grant Nos.60604033, 60553001,and the Hi-Tech research and Development Program of China Grant No.2006AA10Z216

8. REFERENCES

[1] E. Agichtein, E. Brill, S. Dumais, and R. Ragno. Learning user interaction models for predicting web search result preferences. In *Proceedings of SIGIR2006*, 2006.

[2] R. Agrawal, A. Halverson, K. Kenthapadi, N. Mishra, and P. Tsaparas. Generating labels from clicks. In *Proceedings of WSDM2009*, 2009.

[3] J. H. Albert and S. Chib. Bayesian analysis of binary and polychotomous response data. *Journal of the American Statistical Association*, 88(422):669–679, 1993.

[4] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, New York, 2006.

[5] B. Carterette and R. Jones. Evaluating search engines by modeling the relationship between relevance and clicks. In *Proceedings of NIPS20*, 2008.

[6] O. Chapelle and Y. Zhang. A dynamic bayesian network click model for web search ranking. In *Proceedings of WWW2009*, 2009.

[7] N. Craswell, O. Zoeter, M. Taylor, and B. Ramsey. An experimental comparison of click position-bias models. In *Proceedings of WSDM2008*, 2008.

[8] G. Dupret and B. Piwowarski. User browsing model to predict search engine click data from past observations. In *Proceedings of SIGIR2008*, 2008.

[9] F. Guo, C. Liu, A. Kannan, T. Minka, M. Taylor, Y. Wang, and C. Faloutsos. Click chain model in web search. In *Proceedings of WWW2009*, 2009.

[10] K. Jarvelin and J. Kekalainen. Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems 20(4), 422-446 (2002)*, 2002.

[11] T. Joachims, L. Granka, B. Pan, H. Hembrooke, and G. Gay. Accurately interpreting clickthrough data as implicit feedback. In *Proceedings of SIGIR2005*, 2005.

[12] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Throry*, 1998.

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

[14] F. Radlinski and T. Joachims. Query chains: learning to rank from implicit feedback. In *Proceedings of KDD2005*, 2005.

[15] M. Richardson, E. Dominowska, and R. Ragno. Predicting clicks: estimating the click-through rate for new ads. In *Proceedings of WWW2007*, 2007.

Appendix: An Efficient Approach to the Computations of (6), (8) and (9)

Suppose we have a Gaussian variable x_i with mean μ_i and variance σ_i^2. We now compute c_{ik}, u_{ik} and v_{ik} mentioned in Section 3.2. If $k = 0$, the computation is trivial. If $k \geq 1$, it is worthy noting that

$$c_{ik}\mathcal{N}(x_i; \frac{u_{ik}}{c_{ik}}, \frac{c_{ik}v_{ik} - u_{ik}^2}{c_{ik}^2}) \tag{15}$$

gives the approximation to

$$\Phi^k(x_i)\mathcal{N}(x_i; \mu_i, \sigma_i^2) \tag{16}$$

with the minimum KL-divergence between both. Thus, if we find a function in the form of (15) which approximates (16) by minimizing the KL-divergence, we can retrieve the values of c_{ik}, u_{ik}, v_{ik} from this function's parameters directly.

We give the computational approach as follow. Essentially, this approach is implemented by first approximating (16) and then retrieving the values we need from (15). The approximation is made by using Expectation Propagation[13] and the iterative message passing algorithm on factor graphs[12]. Several intermediate variables μ_A, σ_A, μ_B, σ_B and d_0, d_1, d_2 are introduced to simplify the representation of formulas. At the beginning of the algorithm, μ_B and σ_B are initialized to be 0 and 1 respectively, if no prior knowledge is available. The following two steps of assignments should be sequentially executed.

Step A

$$\mu_A \leftarrow \frac{(k-1)\sigma_i^2\mu_B + \sigma_B^2\mu_i}{(k-1)\sigma_i^2 + \sigma_B^2},$$
$$\sigma_A \leftarrow \sqrt{\frac{\sigma_i^2\sigma_B^2}{(k-1)\sigma_i^2 + \sigma_B^2} + 1}.$$

Step B

$$d_0 \leftarrow \Phi\left(\frac{\mu_A}{\sigma_A}\right),$$
$$d_1 \leftarrow \mu_A + \frac{\sigma_A}{\sqrt{2\pi}d_0}\exp\left(-\frac{\mu_A^2}{2\sigma_A^2}\right),$$
$$d_2 \leftarrow \sigma_A^2 + \mu_A d_1 - d_1^2,$$
$$\mu_B \leftarrow \frac{\sigma_A^2 d_1 - d_2\mu_A}{\sigma_A^2 - d_2},$$
$$\sigma_B \leftarrow \sqrt{\frac{d_2\sigma_A^2}{\sigma_A^2 - d_2} + 1}.$$

If $k = 1$, the above assignments are performed only once. If $k \geq 2$, Step A and Step B should be iteratively executed for several times until μ_B and σ_B do not change any more. Then we have

$$S = \frac{d_0}{\mathcal{N}(\mu_A; \mu_B, \sigma_A^2 + \sigma_B^2 - 1)},$$
$$c_{ik} = \frac{S^k \cdot \exp\left(-\frac{k(\mu_B - \mu_i)^2}{2\sigma_B^2 + 2k\sigma_i^2}\right)}{\sqrt{(2\pi)^k \cdot \sigma_B^{2k-2}(\sigma_B^2 + k\sigma_i^2)}},$$
$$u_{ik} = c_{ik}\frac{k\sigma_i^2\mu_B + \sigma_B^2\mu_i}{k\sigma_i^2 + \sigma_B^2},$$
$$v_{ik} = c_{ik}\frac{\sigma_i^2\sigma_B^2}{k\sigma_i^2 + \sigma_B^2} + \frac{u_{ik}^2}{c_{ik}}.$$

After the computation completes, we store the values of μ_B and σ_B. Thus at the next time when c_{ik}, u_{ik} and v_{ik} are evaluated, μ_B and σ_B can be initialized by the stored values.