Extracting Search Query Patterns via the Pairwise Coupled Topic Model

Takuya Konishi^{†§}
takuya-ko@nii.ac.jp

Takuya Ohwa^{†§}
takuyaohwa@nii.ac.jp

Sumio Fujita[‡]
sufujita@yahoo-corp.jp

Kazushi Ikeda^{*}
kazushi@is.naist.jp

Kohei Hayashi^{†§}
kohei-h@nii.ac.jp

[†]National Institute of Informatics [§]JST, ERATO, Kawarabayashi Large Graph Project

[‡]Yahoo Japan Corporation ^{*}Nara Institute of Science and Technology

ABSTRACT

A fundamental yet new challenge in information retrieval is the identification of patterns behind search queries. For example, the query “NY restaurant” and “boston hotel” shares the common pattern “LOCATION SERVICE”. However, because of the diversity of real queries, existing approaches require data preprocessing by humans or specifying the target query domains, which hinders their applicability.

We propose a probabilistic topic model that assumes that each term (e.g., “NY”) has a topic (LOCATION). The key idea is that we consider topic co-occurrence in a query rather than a topic sequence, which significantly reduces computational cost yet enables us to acquire coherent topics without the preprocessing. Using two real query datasets, we demonstrate that the obtained topics are intelligible by humans, and are highly accurate in keyword prediction and query generation tasks.

Keywords

Search queries; Query intent analysis; Topic model

1. INTRODUCTION

When people want to find new information on the Internet, they commonly use search engines. Users interact with search engines by queries, and their intents and personal backgrounds are expected to be preserved in query logs.

Extracting the low-dimensional hidden structures behind search queries is an essential task to obtain user information. This is motivated by our intuition that, while real queries are diverse and unorganized and the number of possible queries are nearly infinite, most real queries fall into a combination of a few terms from multiple categories. For example, the query “NY restaurant” consists of two terms “NY” and “restaurant”. Clearly, “NY” indicates LOCATION and “restaurant” specifies the type of SERVICE. If we know such *is-a* relationships, we can infer that the user wants to

use SERVICE and s/he lives in/will go to LOCATION in the near future. Such knowledge provides a rich interpretation about user needs and is beneficial for better search experiences in related applications. Query recommendation in Internet advertising is a typical example [2, 15]. For advertising clients, extracted query patterns can be used to suggest cost-effective queries, i.e., less-known and having low advertising rate but expected to have the same advertising performance as hot keywords. Furthermore, interpretable query patterns are potentially applicable to predicting click through rate [11], profiling users [9], improving search results [17], and query expansion [21].

The identification of such categories and *is-a* relationships has emerged as a new area in information retrieval (IR). This is referred to as query templates [1] and several approaches has been proposed to solve this task [1, 8, 14]. Although such approaches have demonstrated the usefulness of query templates for real applications, they have several limitations. First, they need the specification of target domain beforehand [1, 14], i.e., we must collect queries that are related to a certain theme, such as automobiles, travels, and movies. This preprocessing reduces the number of queries and the diversity drastically. However, it loses the generality and much useful information, such as cross-domain knowledge. Another issue is that the need for human assistance [8]. While humans can identify the term-category relationships with high accuracy, its expensive resource cost and low throughput hampers the applicability to large-scale query logs. In addition, due to the expensive computational cost, the number of the category (K) is considerably restricted to be small such as $K \simeq 6$.

Topic models [3, 4, 22] are alternatives to solve the query template task. Topic models learn the relationships between a term and a category as a *topic* from a large number of documents (i.e., queries) in an unsupervised manner. Desirably, topic models are feasible with $K \geq 100$ categories. However, they typically assume that each document contains many terms and are not suitable for query logs, which are extremely sparse [3, 4]. In addition, there is no straightforward way to incorporate information from a sequence of topics. For example, a topic sequence (LOCATION, SERVICE) appears frequently in real queries, but (SERVICE, SERVICE) is very rare. Exploiting such correlation information will further improve the performance.

To address the above issues, we propose the pairwise coupled topic model (PCTM)—a probabilistic topic model for

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).

WSDM’16, February 22–25, 2016, San Francisco, CA, USA.

© 2016 Copyright held by the owner/author(s).

ACM ISBN 978-1-4503-3716-8/16/02.

DOI: <http://dx.doi.org/10.1145/2835776.2835794>.

query logs. The PCTM approximates the co-occurrence information over queries by topic pairs, which significantly enriches the model and overcomes the brevity and sparseness of queries. On the basis of the approximation, we derive a fully-Bayesian inference algorithm with collapsed Gibbs sampling, which handles queries as a collection of term pairs and significantly reduces computational cost from $O(K^M)$ to $O(M^2K^2)$ where M is the length of a query. In the PCTM, M is essentially very small and we can manage sufficient sizes of K as $K \simeq 100$. Moreover, by exploiting the sparsity of the topic correlation, we develop an efficient computation technique for the collapsed Gibbs sampling.

Our contributions are summarized as follows.

Versatility: The PCTM can handle queries of any domains, and it is not necessary to specify them. In addition, the PCTM does not require human assistance.

Extracting sparse cross-domain relation: The PCTM estimates topic co-occurrence as a sparse covariance matrix, which gives us cross-domain knowledge as an interpretable network of topics.

Computational Efficiency: Our computational trick reduces the sampling cost from $O(K^2)$ to $O(K)$ with almost no expense.

Our study primarily focuses on extracting the structures that commonly appear among search queries, rather than adapting specific information retrieval tasks. Thus, we evaluate the PCTM with respect to the following three fundamental viewpoints: the coherence of estimated topics, the accuracy of keyword prediction where the model infer a term given other terms in the query, and the ability of query generation. These abilities are key in many applications. For example, coherent topics help practitioners to understand the results of specific IR tasks in a deeper way. Keyword prediction is used for query suggestion and spelling correction. Generating natural queries is necessary for query recommendation in Internet advertising.

2. MODELING QUERY LOGS

2.1 Problem Definition

Given a set of queries, we would like to obtain the following knowledge: (a) a set of distinct categories of terms used in query logs, (b) *is-a* relationships between the categories and terms, and (c) a mapping from a query to query pattern represented by a sequence of the categories. Let us explain the detail by using an example. Suppose we have the query, “chicago hotel cheapest” where “chicago”, “hotel”, and “cheapest” are instances of the categories **LOCATION**, **SERVICE**, and **CONDITION** respectively. Such categories are related to many other terms, e.g., **LOCATION** is associated with a variety of terms that indicate the names of places. Note that a term can possibly belong to multiple categories, e.g., “chicago” may be associated with a **MOVIE** category in other queries. With such many-to-many relationships, the query pattern is recovered as a sequence of categories (**LOCATION**, **SERVICE**, **CONDITION**).

We notice that it is difficult to obtain the category information from any other external resources in advance (e.g., dictionaries and thesauruses) because the concepts defined by external resources do not always correspond to desirable categories that reflect the actual query activities. For example, both “houston” and “miami” are names of places; however, queries containing “houston” and queries contain-

ing “miami” could be very different in terms of user intent and personal backgrounds. The former is in a residential area, and would likely be used with **DAILYLIFE** keywords, such as “houston apartment” or “houston job”. In contrast, the latter is in a resort area and would likely be used with **TRAVEL** keywords, such as “miami hotel” or “miami restaurant”. Treating these two terms as the same category loses such query-specific information.

2.2 Naive Query Model

To obtain knowledge from query logs, we consider a simple probabilistic model of a query based on a topic model. Suppose we have an M -long query $\mathbf{q} = (w_1, \dots, w_M)$ where each term $w_m \in \{1, \dots, V\}$ has a latent topic $z_m \in \{1, \dots, K\}$ and these terms are conditionally independent given the topics. Under this assumption, the joint probability of $\mathbf{q}$ and $\mathbf{z} = (z_1, \dots, z_M)$ is expressed as follows:

$$p(\mathbf{q}, \mathbf{z}) = p(\mathbf{z})p(\mathbf{q}|\mathbf{z}) \\ = p(z_1, \dots, z_M) \prod_{w_m \in \mathbf{q}} p(w_m|z_m). \quad (1)$$

This formulation satisfies the requirements mentioned in Section 2.1: (a) categories (i.e., topics) are represented as the latent variables $\mathbf{z}$, (b) *is-a* relationships between w_m and z_m are represented as term distribution $p(w_m|z_m)$ that indicates the probability of term occurrences given a topic, and (c) a sequence of the categories is represented as a topic sequence $(z_1, \dots, z_M)$, and $p(z_1, \dots, z_M)$ is the distribution that indicates the probability of the occurrences of the topic sequence. Note that representing *is-a* relationships by the term distribution enables that one term can be associated with multiple topics.

Without loss of generality, model (1) is rewritten as a product of the multinomial distribution for terms and the tensor-variate multinomial distribution for a topic sequence as follows:

$$p(\mathbf{q}, \mathbf{z}) = \pi_{z_1, \dots, z_M} \prod_{w_m \in \mathbf{q}} \phi_{w_m, z_m}. \quad (2)$$

ϕ_{w_m, z_m} denotes the probability of the term w_m given topic z_m , and $\pi_{z_1, \dots, z_M}$ denotes the probability of the topic sequence $(z_1, \dots, z_M)$; thus, $\pi_{z_1, \dots, z_M}$ can be considered as an M -th order tensor having K^M elements, which represent the probabilities of all possible sequences of topics $z_1, \dots, z_M$.

Model (2) is general and flexible; however, it requires huge computational cost in parameter estimation. Bayesian inference needs to marginalize the latent variables $\mathbf{z}$, which is equivalent to computing a summation of all possible values of $\pi_{z_1, \dots, z_M}$ in Eq. (2) and this requires $O(K^M)$ computation. Even though query length M is relatively small compared to that of normal documents, practically M could be more than 10 in real query logs and it would easily make parameter estimation computationally infeasible.

3. PCTM: PAIRWISE COUPLED TOPIC MODEL

As discussed above, the main computational cost of model (2) arises from the full dependency of $p(z_1, \dots, z_M)$. In this section, we consider relaxation of this assumption.

3.1 Pairwise Decomposition of Topics

Our key idea is that, in most queries, the full dependency of a topic sequence is sufficiently approximated by a col-

lection of pairwise relationships among topics. For example, let us reconsider query “chicago hotel cheapest”. Here, we can recognize that “chicago” is used as **LOCATION** rather than **MOVIE** because it is jointly used with “hotel”. Similarly, “hotel cheapest” sufficiently determines the context of both “hotel” and “cheapest”. Therefore, in this example, two term pairs “chicago hotel” and “hotel cheapest” are sufficient to estimate the intent of this query.

By following this observation, we assume that $p(z_1, \dots, z_M)$ is decomposed as the product of the second-order tensors with respect to z . For example, if $M = 3$, the joint probability of topics is written as:

$$p(z_1, z_2, z_3) \propto (\Psi(z_1, z_2)\Psi(z_1, z_3)\Psi(z_2, z_3))^{\frac{1}{2}}, \quad (3)$$

where $\Psi(\cdot, \cdot)$ is a potential function representing a pairwise interaction of topics. Note that we introduce exponential weight $\frac{1}{M-1} = \frac{1}{2}$ for later convenience. While a general potential function takes unnormalized and unconstrained real values, we suppose the each potential indicates the probabilities of the occurrence of topic pairs. In spite of such definition, Eq. (3) requires normalizing constant because of dependencies between potentials. Eq. (3) can be generalized as follows:

$$p(z_1, \dots, z_M) \propto \left(\prod_{(i,j) \in B_M} \Psi(z_i, z_j) \right)^{\frac{1}{M-1}}, \quad (4)$$

where B_M is the set of all possible pair indices in an M -long sequence, i.e., $|B_M| = {}_M C_2 = M(M-1)/2$. Substituting decomposition (4) into model (1) yields:

$$\begin{aligned} p(\mathbf{q}, \mathbf{z}) &\propto \prod_{w_m \in \mathbf{q}} p(w_m | z_m) \left(\prod_{(i,j) \in B_M} \Psi(z_i, z_j) \right)^{\frac{1}{M-1}} \\ &\propto \left(\prod_{(i,j) \in B_M} p(w_i | z_i) p(w_j | z_j) \Psi(z_i, z_j) \right)^{\frac{1}{M-1}}. \end{aligned} \quad (5)$$

Interestingly, Eq. (5) implies that pairwise decomposition of topics (3) allows us to deal with the entire model as the product of the sub model with respect to term pairs (w_i, w_j) and topic pairs (z_i, z_j) for $(i, j) \in B_M$.

3.2 Modeling Terms and Topic Pairs

Model (5) yet has an issue of the computational cost. The parameters of the potentials depend on the normalizing constant, which requires $O(K^M)$ computation.

To avoid this, we decompose queries into term pairs and directly model the term pairs. We introduce new variables: $\mathbf{b} = \{b_i\}_{i=1}^{|B_M|} = \{(w_{i,1}, w_{i,2})\}_{i=1}^{|B_M|}$ denotes all possible term pairs in an M -long query and $\mathbf{x} = \{x_i\}_{i=1}^{|B_M|} = \{(z_{i,1}, z_{i,2})\}_{i=1}^{|B_M|}$ denotes the corresponding topic pairs. Differently from Eq. (5), all the term and topic pairs are defined as different variables. The new model is then described as

$$p(\mathbf{b}, \mathbf{x}) = \prod_{i=1}^{|B_M|} p(b_i | x_i) p(x_i), \quad (6)$$

where $p(b_i | x_i) = p(w_{i,1} | z_{i,1}) p(w_{i,2} | z_{i,2}) = \phi_{w_{i,1}, z_{i,1}} \phi_{w_{i,2}, z_{i,2}}$, $p(x_i) = p(z_{i,1}, z_{i,2}) = \psi_{z_{i,1}, z_{i,2}}$.

$p(x_i)$ is the $K \times K$ matrix-variate multinomial distribution where the parameter $\psi_{\cdot, \cdot}$ represents the probability of topic co-occurrence, i.e., the covariance of topics. As in Eq. (1),

this model satisfies requirements (a) and (b) described in Section 2.2. We will discuss (c) in Section 3.4.

Compared to model (5), model (6) drastically reduces the computational cost. Since all latent variables are independent in $p(\mathbf{b}, \mathbf{x})$, we can separately take the marginalization with respect to topic pairs, which reduces computational cost from the topic sequence $O(K^M)$ to the sum of topic pairs $O(M^2 K^2)$.

On the basis of model (6), we construct the PCTM for the set of queries. Suppose we have $N \geq M$ term pairs obtained by decomposing M queries.¹ Let $B = \{b_i\}_{i=1}^N = \{(w_{i,1}, w_{i,2})\}_{i=1}^N$ denote the term pairs and $X = \{x_i\}_{i=1}^N = \{(z_{i,1}, z_{i,2})\}_{i=1}^N$ denote the corresponding topic pairs. By introducing the Dirichlet prior distributions $p(\phi_k | \beta) \equiv \text{Dir}(\phi_k | \beta \mathbf{1})$ ($k = 1, \dots, K$) and $p(\psi | \gamma) \equiv \text{Dir}(\psi | \gamma \mathbf{1})$, the joint distribution of the PCTM is given as

$$\begin{aligned} p(B, X, \phi, \psi | \beta, \gamma) &= \prod_{i=1}^N p(b_i | x_i, \phi) p(x_i | \psi) \prod_{k=1}^K p(\phi_k | \beta) p(\psi | \gamma) \\ &= \prod_{i=1}^N p(w_{i,1} | z_{i,1}, \phi) p(w_{i,2} | z_{i,2}, \phi) p(x_i | \psi) \prod_{k=1}^K p(\phi_k | \beta) p(\psi | \gamma), \end{aligned} \quad (7)$$

where $\phi = \{\phi_k\}_{k=1}^K$.

3.3 Inference Algorithm

We infer the parameters by collapsed Gibbs sampling that enable Bayesian inference, which is more robust and effective for avoiding overfitting than point estimation by e.g. the EM algorithm. The joint distribution after marginalizing out ϕ and ψ is given as:

$$\begin{aligned} p(B, X | \beta, \gamma) &= \int p(B | X, \phi) p(\phi | \beta) d\phi \int p(X | \psi) p(\psi | \gamma) d\psi \\ &= \left(\frac{\Gamma(V\beta)}{\Gamma(\beta)^V} \right)^K \prod_{k=1}^K \frac{\prod_{v=1}^V \Gamma(n_{v|k} + \beta)}{\Gamma(n_{\cdot|k} + V\beta)} \\ &\quad \frac{\Gamma(K^2\gamma)}{\Gamma(\gamma)^{K^2}} \frac{\prod_{k=1}^K \prod_{l=1}^K \Gamma(n_{k,l} + \gamma)}{\Gamma(N + K^2\gamma)}, \end{aligned} \quad (8)$$

where $\Gamma(\cdot)$ is the gamma function, $n_{v|k} = \sum_{i=1}^N \mathbb{I}_v[w_{i,1}] \mathbb{I}_k[z_{i,1}] + \mathbb{I}_v[w_{i,2}] \mathbb{I}_k[z_{i,2}]$, $n_{\cdot|k} = \sum_{v=1}^V n_{v|k}$, and $n_{k,l} = \sum_{i=1}^N \mathbb{I}_k[z_{i,1}] \mathbb{I}_l[z_{i,2}]$ ($\mathbb{I}_a[b]$ is an indicator function that takes 1 when $a = b$). Note that the marginalization has the closed-form solution as in Eq. (8) because of the inconsistency of the topics introduced in Eq. (6). This makes expensive sampling of ϕ and ψ unnecessary, whereas original model (5) requires.

Samples of X are obtained by the following conditional distribution:

$$\begin{aligned} p(x_i = (k, l) | B^{-i}, b_i = (w_{i,1}, w_{i,2}), X^{-i}) \\ \propto \begin{cases} (n_{k,l}^{-i} + \gamma) \frac{(n_{w_{i,1},1|k}^{-i} + \beta)(n_{w_{i,2},2|l}^{-i} + \beta)}{(n_{\cdot|k}^{-i} + V\beta)(n_{\cdot|l}^{-i} + V\beta)} & (k \neq l) \\ (n_{k,k}^{-i} + \gamma) \frac{(n_{w_{i,1},1|k}^{-i} + \beta)(n_{w_{i,2},2|k}^{-i} + \beta)}{(n_{\cdot|k}^{-i} + 1 + V\beta)(n_{\cdot|k}^{-i} + V\beta)} & (k = l) \end{cases}, \end{aligned} \quad (9)$$

¹Hereafter, we ignore single-term queries so that each query contains two terms or more. We also assume that a query does not include the same terms; thus, we eliminate duplicate terms, e.g., we convert “NY NY restaurant” to “NY restaurant”.

Algorithm 1 Collapsed Gibbs sampling for the PCTM

```
for all term pairs  $i = 1 \dots N$  do
  for all topic pairs  $(k, l) = (1, 1) \dots (K, K)$  do
    Compute the probability  $p(x_i = (k, l) | B, X^{-i})$ 
  end for
  Sample the topic pair  $x_i$ 
end for
```

where the notation $-i$ indicates the set of variables or the counts excluding the variable at the i th position. We compute Eq. (9) for all term pairs as summarized in Algorithm 1.

The mechanism of learning topics with the PCTM may not be intuitive. While other topic models (e.g., latent Dirichlet allocation (LDA) [4]) learn topics by term co-occurrence, the PCTM does not use this information explicitly in the modeling level. However, the PCTM does use this information during inference. This is why the PCTM can learn coherent topics. By following Algorithm 1, let us explain this with a simple example. Suppose we observe a term pair $b_1 = (\text{"chicago"}, \text{"hotel"})$ as a first data query. Since $n_{w|k}$ and $n_{k,l}$ are initialized randomly, the topics of "chicago" and "hotel" are sampled randomly. Let "chicago" and "hotel" be assigned to topic 1 and topic 2 respectively. These assignments increase $n_{\text{chicago}|\text{topic1}}$, $n_{\text{hotel}|\text{topic2}}$, and $n_{\text{topic1},\text{topic2}}$. Next, suppose we observe a term pair $b_2 = (\text{"ny"}, \text{"hotel"})$. Here, "hotel" will be assigned to topic 2 because $n_{\text{hotel}|\text{topic2}}$ is higher. Now, "ny" will be assigned to topic 1 because, while $n_{\text{ny}|\text{topic1}}$ and $n_{\text{ny}|\text{topic2}}$ are still random, $n_{\text{topic1},\text{topic2}}$ is higher. In this example, although "chicago" and "ny" are never co-occurred in b_1 and b_2 , they are assigned to the same topic 1.

We estimate the integrated ϕ and ψ as follows:

$$\hat{\phi}_{v,k} = \frac{n_{v|k} + \beta}{n_{\cdot|k} + V\beta} \quad \text{and} \quad \hat{\psi}_{k,l} = \frac{n_{k,l} + \gamma}{N + K^2\gamma}, \quad (10)$$

where $n_{v|k}$, $n_{\cdot|k}$ and $n_{k,l}$ are computed by the obtained samples of topic pairs from Algorithm 1.

3.4 Recovering Topic Sequences with Topic Inconsistency

As mentioned, changing formulation from Eq. (5) to Eq. (6) improves computational complexity. However, Eq. (6) loses information about queries, and it causes the *topic inconsistency* problem. Given a query "chicago hotel cheapest", consider two term pairs $b_1 = (\text{"chicago"}, \text{"hotel"})$ and $b_2 = (\text{"chicago"}, \text{"cheapest"})$. In Eq. (5), $z_{1,1}$ and $z_{2,1}$ are treated as the same variable. However, in Eq. (6), they are different variables, and the estimated topics can be different. This problem becomes an obstacle for satisfying the requirement (c) of Section 2.1.

To avoid this, we consider an approximate way to recover the most probable topic sequence. Consider an M -long query $\mathbf{q} = (w_1, \dots, w_M)$ with a topic sequence $\mathbf{z} = (z_1, \dots, z_M)$. In the PCTM, the posterior of the topic pair x_i given the term pair b_i and the estimated parameters $\hat{\phi}$ and $\hat{\psi}$ is written as follows:

$$p(x_i | b_i, \hat{\phi}, \hat{\psi})_{kl} = \frac{\hat{\psi}_{k,l} \hat{\phi}_{w_{i,1},k} \hat{\phi}_{w_{i,2},l}}{\sum_{k'=1}^K \sum_{l'=1}^K \hat{\psi}_{k',l'} \hat{\phi}_{w_{i,1},k'} \hat{\phi}_{w_{i,2},l'}}. \quad (11)$$

By combining with Bayes' rule, we approximate the poste-

Figure 1: Estimated $\hat{\psi}$ from the AOL dataset (see Section 5 for more details)

rior of the topic sequence by the following Markov chain:

$$\begin{aligned} p(\mathbf{z} | \mathbf{q}) &\approx p(z_1, z_2 | \mathbf{q}) p(z_3 | z_2, \mathbf{q}) p(z_4 | z_3, \mathbf{q}) \dots p(z_M | z_{M-1}, \mathbf{q}) \\ &\approx p(x_1 | b_1) p(x_2 | b_2) p(x_3 | b_3) \dots p(x_{M-1} | b_{M-1}) \end{aligned} \quad (12)$$

where

$$p(x_{i|i-1} | b_i)_{kl} = \frac{p(x_i | b_i)_{kl}}{p(z_{i-1,1} | b_i)_l} = \frac{p(x_i | b_i)_{kl}}{\sum_k p(x_i | b_i)_{kl}} \quad (13)$$

denotes the conditional posterior of $z_{i,2}$ given $z_{i,1}$. The last line of Eq. (12) enables us to compute the most probable sequence of topics by forward algorithms, such as the Viterbi algorithm [18].

3.5 Speeding-up Gibbs Sampling

Collapsed Gibbs sampling (Algorithm 1) requires $O(K^2)$ computations to check the probabilities of each topic pair, and the total complexity amounts to $O(NK^2)$. While this is significantly better than that of the naive model (1), the K^2 term is sometimes problematic when K is large.

Surprisingly, by exploiting the property of query logs, we can reduce the $O(K^2)$ computation to $O(K)$. The key assumption is that, in a query, a topic co-occurs with only a few specific topics. For example, a topic **SERVICE** co-occurs frequently with **LOCATION**, but rarely with other topics such as **MOVIE**, **ELECTRONICS** and **TOY**. Moreover, if the number of frequently co-occurred topics is $O(1)$ for each topic, the total number of meaningful topic pairs is bounded by $O(K)$. In such case, estimated $\hat{\psi}$ is supposed to be very sparse. In fact, as demonstrated in Figure 1, we observe that this assumption is fairly reasonable for real query logs.

On the basis of the above property, we propose a fast sampling algorithm that skips unnecessary probability computations in the collapsed Gibbs sampling,² which we refer to as *conditioned sampling*. First, we define that topic pair (k, l) is *active* if $n_{k,l} > 0$ in current samples; otherwise *inactive* (i.e., $n_{k,l} = 0$). Then, we consider to divide topic pairs into the two sets S and $\bar{S}$; S is the set of a few active topic pairs having high probability, and $\bar{S}$ is the set of many other inactive topic pairs having low probability. Let μ denote the total probability of the active topic pairs.

Now we consider to sample x_i from Eq. (9). For each topic pair x_i , we flip a biased coin such that $p(\text{head}) = \mu$. If it comes up heads (tails), we sample x_i by computing Eq. (9) among only S ($\bar{S}$). In most cases, x_i is sampled

²Similar techniques are used in [23].

Algorithm 2 Conditioned sampling for the PCTM

```
for all term pairs  $i = 1 \dots N$  do
  Compute the probability  $\mu$ 
  Choose  $S$  or  $\bar{S}$  according to  $\mu$ 
  if Select  $S$  then
    for all active topic pairs  $(k, l)$  in  $S$  do
      Compute the probability  $p(x_i = (k, l) | B, X^{-i})$ 
    end for
    Sample the topic pair  $x_i$ 
  else
    for all inactive topic pairs  $(k, l)$  in  $\bar{S}$  do
      Compute the probability  $p(x_i = (k, l) | B, X^{-i})$ 
    end for
    Sample the topic pair  $x_i$ 
  end if
end for
```

from S because μ is larger than $1 - \mu$. In addition, sampling from S is more tractable than the exact sampling because we compute Eq. (9) for only a few active topic pairs. Conversely, sampling from $\bar{S}$ requires $O(K^2)$; however because $1 - \mu$ is sufficiently small, this procedure rarely occurs. Taken together, the average time complexity per x_i is $O(\mu|S| + (1 - \mu)(K^2 - |S|))$. Here we assume two conditions: (i) $|S| = O(K)$, which we discussed above, and (ii) $1 - \mu = O(1/K)$. Under these conditions, $\mu|S| = O(K)$ and $(1 - \mu)(K^2 - |S|) = O(K)$, and the total complexity is reduced to $O(K)$. Whole the procedures are summarized as in Algorithm 2. Now we check the validity of condition (ii). Substituting zero into all the counts such as $n_{k,l}^{-i}$ in Eq. (9) yields $p(x_i = (k, l) | B, X^{-i}) \propto \gamma$ (we ignore $1/V^2$ because it is constant), which implies that $1 - \mu$ has $O(K^2\gamma)$ probability. Because its normalizer is approximately given by $N + K^2\gamma$ (see the update equation of ψ (10)), $1 - \mu = O(1/K)$ unless $N > K^3\gamma$. Here, we empirically found that the PCTM with the small γ such as $O(1/K^2)$ performs well. Instead, as long as using such small γ , the condition is relaxed to $N > K$, which is obviously satisfied.

Note that, if we know the true value of $\mu = \sum_i \sum_{(k,l) \in S} p(x_i = (k, l) | B, X^{-i})$, the obtained samples from the above procedure are exactly the same as the original sampler. However, the computation of μ requires the summation of Eq. (9), taking $O(NK^2)$. Therefore, as an approximation, we use the following probability:

$$\mu = \sum_{(k,l) \in S} p(x_i = (k, l) | X^{-i}) = \frac{N - 1 + |S|\gamma}{N - 1 + K^2\gamma}, \quad (14)$$

which we can compute in constant time. Note that Eq. (14) is the total probability of the active topic pairs with assuming that B is generated by the model, instead of the observed B . Therefore, it may have a small deviation from the true μ ; however, we empirically confirmed that the obtained samples have almost the same quality as the exact ones (Section 5.6).

4. RELATED WORK

4.1 Topic Models for Search Logs

Topic models have been widely used for modeling search log datasets with different motivations. One motivation has

been to improve search results based on search log histories [5, 9, 17]. For example, Harvey *et al.* proposed a topic model that incorporates search engine users for user profiling to improve personalized searches [9]. Other researchers have explored topic models for traditional search log tasks, such as predicting click-through rates [11, 12]. Jiang *et al.* proposed a topic model for some information included in search logs, such as URLs and timestamps [11]. Search logs are valuable sources of information for obtaining web-based knowledges. Xu *et al.* proposed a topic model for named entity mining with an efficient semi-supervised learning algorithm [20].

As in the above models, topic models have been used for modeling search log datasets that contain not only queries and other information, such as users, time stamps, URLs, and click-through logs. On the other hand, we focus on modeling the queries themselves. In our setting, we require only raw queries, which extends the range of application.

4.2 Query Templates

Query template methods have been explored to obtain query patterns [1, 8, 14]. Agarwal *et al.* advocated concepts of query templates and proposed a probabilistic inference framework for mining templates based on tripartite graphs among queries, websites, and templates [1]. Han *et al.* studied a human-assisted method for analyzing query templates that incorporates crowdsourcing for query interpretation [8].

Pandey and Punera proposed a probabilistic generative model for queries and query templates [14]. In that model, templates are defined as a topic sequence without duplication, which is not an actual topic assignment for each term. Given a template per query, the model generates the number of terms assigned to the topics included in the template with the Poisson distribution and assigns the topics for each term. For example, the query “toyota 2002 seat cover” is assumed to have a template (BRAND, YEAR, PARTS) and is generated by the topic assignments (BRAND, YEAR, PARTS, PARTS), that is, the model generates “toyota” from BRAND, “2002” from YEAR, and “seat” and “cover” from PARTS.

Although these methods have been demonstrated to extract meaningful query patterns, they only dealt with queries in a particular domain. This restriction improves the quality of query templates for that domain; however, it must perform preprocessing to limit target domains by using such as query classification. For example, one proposed method [1] assumes that *is-a* relationships in a domain are given *a priori* using other methods, such as named entity mining, and they only focused on query pattern extraction. Another proposed model [14] does not employ assumptions about domains; however, it also requires domain limitation due to the computational cost that arises from the dependencies among topics. Compared to those methods, we tackle a more challenging task, i.e., extraction of category and query patterns simultaneously without domain limitation and other preprocessing methods.

4.3 Relation to Other Topic Models

The PCTM is related to other topic models, including LDA [4], the most popular topic model. We can apply LDA to query logs by considering a single query as a document; however the performance may not be good due to the following reasons. First, although LDA assumes that a topic assigned to a term is drawn from a document-specific topic

Figure 2: Generative processes of (a) BTM and (b) PCTM

distribution, short texts such as queries have no more than a few terms in a document. This would cause the sparsity problem, i.e., LDA suffers from the lack of information for each query in estimating the topic distribution [22]. Moreover, if a dataset contains D documents, LDA has a parameter of size $D \times K$, and since D is normally very large in query logs, we must consider memory usage for parameter inference carefully. In contrast, the PCTM has only one distribution that generates topic pairs over the dataset. This avoids the sparsity problem, and the topic pair distribution is represented by a K^2 matrix, which does not depend on the number of documents. In the parameter inference, LDA requires $O(K)$ computation to check the probability of a single term. Therefore, if the average length of documents is L , the total time complexity is $O(DLK)$.

Yan *et al.* proposed the biterm topic model (BTM) [22]. While the BTM and PCTM can be considered topic models for term pairs, the key difference is that the BTM assumes that a term pair shares the same topic. This assumption leads a great success of the BTM in application to short texts because term pairs in general short texts would rarely take completely different topics. However this assumption is not suitable for modeling queries because a query consists of multiple terms having a variety of topics.

To clarify this difference, we illustrate the generative processes of the BTM and PCTM in Figure 2. Recall the example “NY restaurant” mentioned in Section 1. The BTM assigns the same topic for every two terms. Thus, because the location names and service names, such as “NY” and “restaurant”, frequently appear in the same queries, the BTM estimates them as an integrated topic of **SERVICE** and **LOCATION**. Such mixing of topics is not desirable in terms of the interpretability. In contrast, the PCTM assigns a topic *pair* for every two terms. This allows to learn **SERVICE** and **LOCATION** as different topics.

The correlated topic model (CTM) has been proposed to extract topics and their correlations from documents [3]. The CTM assumes the logistic normal distribution to model the correlation and was shown to be successful meaningful topic correlation from the academic journal dataset. However, since the CTM also models topic distribution for each document, it would suffer from the sparsity problem as discussed in LDA. Moreover, while a variational inference algorithm has been proposed, there is no analytical solution and it requires to perform the conjugate gradient method for

Table 1: Summary of the query log datasets where D is the number of documents, V is the number of terms in the vocabulary, and M is the number of terms in a query.

	D	V	Average M	Language
Yahoo	996K	15K	2.31	Japanese
AOL	583K	14K	2.77	English

the parameter inference, which uses huge amount of memory space.

Note that the PCTM has the same structure as the product space mixture model (PMM) [10]. Compared to the PMM, we provide the following three contributions: we associate the PMM with a fully-dependent topic model (2), we demonstrate applicability to query log modeling, and we formulate a fully-Bayesian framework and derive an efficient inference algorithm with collapsed Gibbs sampling rather than an annealed EM algorithm [10].

5. EXPERIMENTS

5.1 Setting

We prepared two real query logs: AOL and Yahoo! Japan datasets. The AOL dataset consists of approximately 20 million queries [16]. The Yahoo! Japan dataset consists of approximately 600 million queries sampled from one week search logs. In our experiments, we examined the abilities of the PCTM for typical queries. To conduct such experiments, we performed the following preprocessing. We eliminated queries from the datasets that included low frequency terms or stop words. We then performed random sampling to reduce the data size. This preprocessing extracted typical queries randomly; while the domains of queries were still diverse. In addition, for the AOL dataset, we performed stemming with the Porter’s algorithm³. Table 1 summarizes the preprocessed datasets.

Throughout the experiments, we compared the PCTM with the LDA and BTM. As explained in Section 4.3, LDA handles a query as a document. Note that we also attempted to evaluate the CTM with an R implementation⁴; however, this did not work in our experimental environment⁵ due to memory overflow. All models were inferred with collapsed Gibbs sampling. We iterated 1,000 samples for the LDA and BTM, and 500 samples for the PCTM. We obtained the last sample in a Markov chain and made use of this sample to estimate parameters, which was required in some experiments. In all experiments and the models, the number of topics was fixed to 100, and β , which is a common hyperparameter, was fixed to 0.1.⁶ Since α of LDA is a sensitive parameter [19], we assumed an asymmetric prior and estimated by using Minka’s fixed point iteration [13]. For α of the BTM and γ of the PCTM, we assumed a symmetric prior and used fixed values; we set α of the BTM to $\frac{50}{K}$ and γ of the PCTM to $\frac{50}{K^2}$ [22].⁷

³<http://tartarus.org/martin/PorterStemmer/>

⁴<http://cran.r-project.org/web/packages/topicmodels/index.html>

⁵CPU: Intel Core i7-3770 3.40GHz, Memory: 16GB RAM

⁶For the LDA and BTM, we follow the notation of hyperparameters from [7] and [22], respectively.

⁷While we did not estimate α of the BTM and γ of the

Figure 3: Results of the word intrusion task. Each panel shows the results for the number of displayed terms.

5.2 Human Evaluation of Topic Quality

In our first experiment, we investigated the interpretability of estimated topics for humans, i.e., how natural and meaningful topics are compared to human knowledge. We conducted three evaluation tasks via crowdsourcing: word intrusion and topic intrusion tasks proposed in [6], and query selection task, which we explain in detail in the following subsections. In these experiments, we used the Yahoo! dataset, and the tasks were performed through using the Yahoo! Japan crowdsourcing service. All jobs contained ten tasks and were assigned to eight different workers [6].

5.2.1 Word Intrusion

The word intrusion task evaluated the cohesion of a group of terms belonging to the same topic. In this task, several terms that belong to the same topic, except for one term (an intruder), were displayed and, a crowdworker attempted to locate the intruder. The intruder was selected randomly at low probability in the topic, but at high probability in another topic. The other displayed terms were selected with high probability in the topic. Therefore, if the topic was significantly cohesive, the intruder was clearly isolated from the other terms, and the crowdworker could find the intruder easily. We performed this task by varying the number of displayed terms (6 and 8). We used 50 topics, from which a large number of terms were assigned in the last samples of the inference. Performance was measured according to the precision of intruders detection. We compute the fraction of intruder detection by 8 workers for each topic.

The results are shown in Figure 3. At first glance, we see that LDA looks worse than the BTM and PCTM. In fact, there was significant difference⁸ between LDA and the BTM, while the difference was not detected between LDA and the PCTM. Although the PCTM was slightly worse than the BTM, the difference was not significant. We discuss the reason why the performances of the BTM and PCTM are nearly the same in Section 6.

5.2.2 Topic Intrusion

The topic intrusion task examined how relevant topics were assigned to queries. For each task, we displayed a query and four topics such that three of the topics were the most relevant to the query but the remaining topic was chosen randomly from the top-10 most irrelevant topics. Then, similar to the word intrusion task, a crowdworker attempted to identify the irrelevant topic. Each topic was represented as T terms having the top- T highest probability in the topic. We

⁸ PCTM, these prior effects will be relatively less than that of LDA because topic (topic pair) distribution is shared among all the latent variables.

⁸ In terms of the one side paired t-test with 95% confidence.

Figure 4: Results of the topic intrusion task

performed this task with varying T (3, 5, and 8). Note that as the number of displayed terms increased, topics became more identifiable. We used 50 queries in total, which were chosen randomly from the dataset. As the same manner of the word intrusion task, we evaluated the performance by the precision. Note that we excluded LDA from this experiment because the estimated topic distributions were very sparse; thus, we could not select three relevant topics.

For the BTM and PCTM, to collect relevant topics for a given query, we estimate topic probability of the query by taking the average of the topic probability of the term pairs. In the BTM, the probability of topic z_q in an M -long query q is expressed as follows:

$$p(z_q|q)_k = \frac{2}{M(M-1)} \sum_{b_i \in B(q)} p(z_i|b_i)_k, \quad (15)$$

where $B(q)$ is the set of term pairs in q and $p(z_i|b_i)_k$ is the posterior probability of topic $z_i = k$ given term pair b_i , which can be estimated in the same manner as Eq. (11). For the PCTM, we first compute the probability of topic pair x_q in query q :

$$p(x_q|q)_{kl} = \frac{2}{M(M-1)} \sum_{b_i \in B(q)} p(x_i|b_i)_{kl}, \quad (16)$$

where $p(x_i|b_i)_{kl}$ is given by Eq. (11). To obtain the probability of individual topic z_q , we then compute the following probability:

$$p(z_q|q)_k = \frac{1}{2} \sum_{l=1}^K p(x_q|q)_{kl} + p(x_q|q)_{lk}, \quad (17)$$

i.e., we take the average for each element and its diagonal element and marginalize out topic l .

The results are shown in Figure 4. We observe that there was no significant difference⁸ between the BTM and PCTM; however, the performances of the PCTM tends to increase as the number of displayed terms increases. This is in contrast to the BTM, in which performance does not change with respect to the number of displayed terms. This implies that the PCTM obtained more coherent topics; thus, crowdworkers could use the increased terms as meaningful information effectively.

5.2.3 Query Selection

The query selection task evaluates how topic models recover a user intent as a query pattern. In each task, we display an actual (target) query and three groups of artificial queries, which are respectively generated by LDA, the BTM, and the PCTM. We let crowdworkers infer the search intent of the target query and then select a group having the most number of queries that have the same intent. For example, we consider a target query as “NY restaurant” in which the intent is assumed to be (LOCATION SERVICE). If a

Figure 5: Results of the query selection task

crowdworker infer the intent correctly, a group consisting of (LOCATION SERVICE)-queries, such as “chicago lottery” and “florida hotel”, should be selected.

We randomly selected 300 2-long queries from the training dataset and used as the target queries. For these 2-long target queries, we estimated the topic pair which takes maximum posterior probability at the end of the inference. For each model, we fixed the length of generated queries to 2 and selected by the following procedure. Given the estimated topic pair in a target query, we compute the posterior probability of 2-long queries, i.e., term pairs. For the PCTM, we computed the probability of term pair $b = (w_1, w_2)$ given topic pair $x = (z_1, z_2)$ and the estimated parameter $\hat{\phi}$ as follows:

$$p(b = (w_1, w_2) | x = (z_1, z_2), \hat{\phi}) = \hat{\phi}_{w_1, z_1} \hat{\phi}_{w_2, z_2}. \quad (18)$$

The BTM and LDA computed the probabilities in the same manner. Finally, we used the top-5 term pairs taking the most high probability as the displayed queries. We measured the score which is the fraction of the selected by 8 workers for each target query.

The results are shown in Figure 5. The PCTM was significantly better⁸ than the BTM and LDA. This means that the PCTM could obtain topic pairs where the intents are closer to ones interpreted by humans than the others.

5.3 Keyword Prediction

In this experiment, we evaluated performance in terms of keyword prediction, i.e., given several terms as a part of a query, we attempted to predict the term that a user will input next. From the whole dataset, we randomly selected 10% of the queries and picked one term for each query as test data. We repeated this procedure 10 times and created training and testing datasets. We randomly initialized the latent variables for each trial. Then, after learning the parameters with the training dataset, we calculated the perplexity for each dataset using the last 100 samples for prediction.

While LDA can perform this experiment simply without modifications, the BTM and PCTM could not. Therefore, we computed the predictive probability for a single term rather than for a term pair. We computed this by assuming that the positions of missing terms in the test queries were known in advance, and that the set of term pairs preserved the missing positions. We regarded missing terms as latent variables in the models and performed collapsed Gibbs sampling on these variables in the parameter inference. The sample of the j th term in the i th term pair was obtained by following the conditional distribution in the PCTM:

$$p(w_{i,j} = v | B^{-i,j}, X^{-i}, x_i = (k, l)) = \frac{n_{w_{i,j}|k}^{-i} + \beta}{n_{\cdot|k}^{-i} + V\beta}, \quad (19)$$

Figure 6: Results of the keyword prediction task. Lower perplexity indicates the better performance.

where $B^{-i,j}$ denotes that the set of term pairs excluding only $w_{i,j}$. The samples of the BTM were obtained in the same manner. After parameter inference, the predictive probability of missing term $w_{i,j}$ was calculated by Eq. (19). We then took the average of the terms, which was contained in separate term pairs but was identical in a query.⁹

The results are shown in Figure 6. For both datasets, the PCTM outperformed the BTM. Indeed, the PCTM was significantly better⁸ than the BTM in both datasets.

5.4 Query Generation

In this experiment, we investigated the ability of the topic models as query generators, which is important for query recommendation application in Internet advertising. To evaluate the performance, we learned each model by using training data and computed the top-10,000 most frequent 2-long queries. Then, we measured AUC against the test queries, which we randomly selected another 100,000 2-long queries¹⁰. In addition, we investigated the total number of generated queries detected in the test data. Note that, while we also performed query generation in the query selection task (Section 5.2.3), the evaluation criterion is different in terms of that the query selection takes the user intent into account; this experiment purely evaluate how the topic models can generate natural queries, regardless of the user intents. We computed the probability of term pair $b = (w_1, w_2)$ whose topic pair is marginalized out as follows:

$$p(b = (w_1, w_2) | \hat{\phi}, \hat{\psi}) = \sum_{z_1} \sum_{z_2} \hat{\phi}_{w_1, z_1} \hat{\phi}_{w_2, z_2} \hat{\psi}_{z_1, z_2}. \quad (20)$$

The probabilities of the BTM and LDA are computed in the same way as the PCTM. Note that we computed the document specific parameters of LDA from estimated α .

The results are shown in Table 2. The AUC of the PCTM was higher than the BTM and LDA in both datasets. This result implies that the occurrence probability of a query computed by the PCTM reflects the occurrence of the real query logs. Moreover, the PCTM detected the most queries from test data in both datasets. The result indicates the PCTM has more natural generative process for real query logs than the BTM and LDA.

5.5 Estimated Topics and Topic Co-occurrences

Here, we compare the obtained topics as a qualitative evaluation using the AOL dataset. We show the LOCATION

⁹Compared to the training dataset of LDA, that of the BTM and PCTM includes the position information about the missing terms. Since LDA does not require such information in the traditional way to compute the perplexity, we did not perform the sampling of the missing terms for LDA.

¹⁰We allowed the test data to include queries consisting of duplicate terms, such as “NY NY” and “restaurant restaurant”.

Table 2: Results of AUC and the total number of detected queries in the query generation task

AUC	Yahoo	AOL	Total num.	Yahoo	AOL
LDA	0.5743	0.5634	LDA	645	632
BTM	0.5729	0.6080	BTM	2,173	1,437
PCTM	0.6021	0.6144	PCTM	3,196	1,880

Table 3: Estimated topics about location

	Top 20 terms
LDA	lottery state universe florida lotto california ny texas unit ohio map michigan job pa georgia result nj illinois virginia service
BTM	san la vegas diego hotel francisco antonio casino california nevada jose nv mission citi cabo reno valley mexico grand lo
PCTM	florida ny texas san nj ohio chicago nc virginia nyc houston fl va pa lo orlando michigan angel la california

topic in Table 3; for each model, we selected a topic such that the most number of terms related to locations in the top-20 topic terms were used. The result shows that the PCTM obtained the most cohesive topic that only contains the terms related to location, while the BTM and LDA topics are corrupted by some non-geographical terms, such as “lottery” and “hotel”. This is a reasonable result for LDA because it counts the topic probability for every document, (i.e., query) and a document is extremely short; thus, terms appearing in the same query were likely to be assigned to the same topic. A similar interpretation holds for the BTM. As described in Section 4.3, the BTM mitigates the brevity problem of queries and assumes that term pairs must be assigned to the same topic, which causes contamination of “not-geographical-but-jointly-used” terms, such as “hotel”.

Figure 1 shows $\hat{\psi}$, which represents the relationships among topics, i.e., $\hat{\psi}_{kl}$ indicates the probability of how often topics k and l are jointly used in term pairs. Note that in Figure 1 the elements are shown in gray where $n_{k,l}$ in Eq. (10) takes 0, which shows that $\hat{\psi}$ is actually very sparse. This indicates that each topic co-occurs with only a few other topics. This result agrees with common sense; people possibly search “NY hotel” but not “lottery hotel”.

An unique advantage of the PCTM compared to the LDA and BTM is that we obtain the relationship among topics as $\hat{\psi}$. Figure 7 represents relationships among “public” topics, which we see that topic 3 represents public facilities and is associated with topics about location (topic 4), public service (topic 0), and job (topic 1). We clearly observe that the PCTM could obtain the network of topics that is very reasonable to our general knowledge. This is a surprising result because the PCTM is a fully unsupervised approach and we did not use any human resources.

5.6 Scalability

In this experiment, we compared the methods with various K as 30, 50, and 100 by using the AOL dataset. First, we checked $|S|$, the size of S , in exact and conditioned sampling. Figure 8 describes the changes of the size in 500 iterations for each setting. We see that, the sizes of both sampling

Figure 7: A subgraph of topic relationships estimated by the PCTM. We put an edge between the topics if the corresponding entry of $\hat{\psi}$ has larger probability than $\frac{\gamma}{N+K^2\gamma}$. For each topic, the top-10 terms are displayed. For each edge, we show the value of $\hat{\psi}$ and the larger value indicates the strong connectivity.

Figure 8: Transitions of the size of S

behaved in surprisingly similar ways. Another interesting observation is that, all the sizes were fluctuated between about 150 and 200, i.e., $|S| = O(1)$ and it suffices condition (i) in Section 3.5.

Next, we evaluated how conditioned sampling reduces the actual runtime. We compared all the models in addition to the PCTM with conditioned sampling. Note that we fixed all the elements of α in LDA to $50/K$ for fair comparisons. The results are shown in Table 4. As expected, LDA, the BTM, and the PCTM with the conditioned sampler run in $O(K)$ time, whereas the normal PCTM required $O(K^2)$.

Finally, we compared exact and conditioned sampling by the keyword prediction task in Section 5.3. The results are shown in Table 5, telling us that conditioned sampling keeps the same performance as exact sampling.

6. CONCLUSION AND DISCUSSION

We have proposed a new probabilistic topic model for query logs. The PCTM can capture topic co-occurrences in a query, which make topics more coherent without limiting the target domain of queries. For model learning, we derived a fully-Bayesian inference algorithm with the efficient computational trick for the collapsed Gibbs sampler. We have examined three types of experiments, i.e., crowd-sourcing, keyword prediction, and query generation tasks.

We discuss four concerns of the PCTM for future improvements.

Firstly, sometimes the PCTM learns overlapped topics,

Table 4: Comparisons of the runtime (in minutes)

	$K = 30$	$K = 50$	$K = 100$
LDA	5.02	7.2	14.31
BTM	8.12	12.84	24.57
PCTM (Exact)	239.79	657.80	2621.02
PCTM (Conditioned)	104.82	153.56	276.50

Table 5: Perplexity of exact and conditioned sampling

	$K = 30$	$K = 50$	$K = 100$
Exact	1451.14	1242.66	1057.96
Conditioned	1450.16	1240.03	1054.38

i.e., the topics having almost the same term distributions. This is caused by the flexibility of the PCTM. Further regularization to remove such redundancy is an important future work of the PCTM.

Secondly, the number of latent variables increases quadratically with query length. Although this situation is fairly better than the approximation of Eq. (5) where the number is the same as query length but the computational cost grows exponentially, it still may cause scaling issues for very long queries. One practical option is just eliminating such long queries before learning the model. This does not affect to the performance significantly because the long queries are quite rare in real datasets.

Thirdly, we discuss that the PCTM did not significantly outperform the BTM in the word and topic intrusion tasks. While the PCTM can obtain separated topic representation as described in Section 4.3, we found that this property did not seem to contribute to the improvement in the both tasks. For example, in the word intrusion task, even though we displayed “hotel” and “florida” included in the integrated topic **LOCATION & SERVICE** estimated by the BTM, crowdworkers possibly infer that these terms are associated with each other more than the other intruder term, because people frequently use location and service names in web search. Conversely, in the query selection task, the PCTM significantly outperformed the BTM because this task requires the separability of topics for what crowdworkers can recognize query intents. In contrast to the word and topic intrusion tasks, the PCTM significantly outperformed the LDA and BTM with respect to keyword prediction, which is a similar characteristic to the CTM; The CTM may select an intruder from highly correlated topics in word intrusion. This results in degenerating the performance because the intruder from the correlated topic confuses crowdworkers [6]. The PCTM seems to have the same problem since it incorporates such correlations between topics.

Fourthly, although the conditioned sampler of the PCTM significantly reduces the computational cost, the runtime is still slower than LDA and the BTM. The main reason is S , the set of active topic pairs, is too large in the initial state. In fact, when $K = 100$, 74% of the runtime was for first 100 iterations. This implies some good initialization for topic pairs may reduce the runtime significantly. For example, the topic distributions obtained by LDA can be useful for the initialization.

Acknowledgments

We thank Naomi Sasaya, Nobuyuki Shimizu, Yoshiko Takeuchi, and Shinichi Tsuzaki for providing helpful comments and organizing the crowdsourcing tasks. KI was supported by MEXT Kakenhi 15H01620. KH was supported by MEXT Kakenhi 15K16055.

7. REFERENCES

- [1] G. Agarwal, G. Kabra, and K. C.-C. Chang. Towards rich query interpretation: Walking back and forth for mining query templates. In *WWW*, 2010.
- [2] R. Agrawal, A. Gupta, Y. Prabhu, and M. Varma. Multi-label learning with millions of labels: Recommending advertiser bid phrases for web pages. In *WWW*, 2013.
- [3] D. M. Blei and J. D. Lafferty. A correlated topic model of science. *The Annals of Applied Statistics*, 1(1):17–35, 2007.
- [4] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet allocation. *JMLR*, 3:993–1022, 2003.
- [5] M. J. Carman, F. Crestani, M. Harvey, and M. Baillie. Towards query log based personalization using topic models. In *CIKM*, 2010.
- [6] J. Chang, J. Boyd-Graber, C. Wang, S. Gerrish, and D. M. Blei. Reading tea leaves: How humans interpret topic models. In *NIPS*, 2009.
- [7] T. L. Griffiths and M. Steyvers. Finding scientific topics. *PNAS*, 101:5228–5235, 2004.
- [8] J. Han, J. Fan, and L. Zhou. Crowdsourcing-assisted query structure interpretation. In *IJCAI*, 2013.
- [9] M. Harvey, F. Crestani, and M. J. Carman. Building user profiles from topic models for personalised search. In *CIKM*, 2013.
- [10] T. Hofmann and J. Puzicha. Statistical models for co-occurrence data. Technical report, Massachusetts Institute of Technology Artificial Intelligence Laboratory, 1998.
- [11] D. Jiang, K. W. T. Leung, W. Ng, and H. Li. Beyond click graph: Topic modeling for search engine query log analysis. In *DASFAA*, 2013.
- [12] D. Jiang and W. Ng. Mining web search topics with diverse spatiotemporal patterns. In *SIGIR*, 2013.
- [13] T. P. Minka. Estimating a Dirichlet distribution. Technical report, Microsoft, 2000.
- [14] S. Pandey and K. Punera. Unsupervised extraction of template structure in web search queries. In *WWW*, 2012.
- [15] S. Pandey, K. Punera, M. Fontoura, and V. Josifovski. Estimating advertisability of tail queries for sponsored search. In *SIGIR*, 2010.
- [16] G. Pass, A. Chowdhury, and C. Torgeson. A picture of search. In *Proceedings of the 1st International Conference on Scalable Information Systems*, 2006.
- [17] W. Song, Y. Zhang, T. Liu, and S. Li. Bridging topic modeling and personalized search. In *COLING*, 2010.
- [18] A. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, 13(2):260–269, 2006.
- [19] H. M. Wallach, D. Mimno, and A. McCallum. Rethinking LDA: Why priors matter. In *NIPS*, 2009.
- [20] G. Xu, S.-H. Yang, and H. Li. Named entity mining from click-through data using weakly supervised latent Dirichlet allocation. In *SIGKDD*, 2009.
- [21] J. Xu and W. B. Croft. Query expansion using local and global document analysis. In *SIGIR*, 1996.
- [22] X. Yan, J. Guo, Y. Lan, and X. Cheng. A biterm topic model for short texts. In *WWW*, 2013.
- [23] L. Yao, D. Mimno, and A. McCallum. Efficient methods for topic model inference on streaming document collections. In *SIGKDD*, 2009.