Named Entity Recognition in Query

Jiafeng Guo[†], Gu Xu[‡], Xueqi Cheng[†], Hang Li[‡]

[†]Institute of Computing Technology, CAS
Beijing, P.R.China
guojiafeng@software.ict.ac.cn,cxq@ict.ac.cn

[‡]Microsoft Research Asia
Beijing, P.R.China
{guxu, hangli}@microsoft.com

ABSTRACT

This paper addresses the problem of Named Entity Recognition in Query (NERQ), which involves detection of the named entity in a given query and classification of the named entity into predefined classes. NERQ is potentially useful in many applications in web search. The paper proposes taking a probabilistic approach to the task using query log data and Latent Dirichlet Allocation. We consider contexts of a named entity (i.e., the remainders of queries after the named entity is removed) as words of a document, and classes of the named entity as topics. The topic model is constructed by a novel and general learning method referred to as WS-LDA (Weakly Supervised Latent Dirichlet Allocation), which employs weakly supervised learning (rather than unsupervised learning) using partially labeled seed entities. Experimental results show that the proposed method based on WS-LDA can accurately perform NERQ, and outperform the baseline methods.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Query formulation*

General Terms

Algorithms, Experimentation

Keywords

Named Entity Recognition, Topic Model

1. INTRODUCTION

In this paper we address a novel problem in web search, namely Named Entity Recognition in Query (NERQ). In the task given a query we are to detect the named entity within the query and identify the most likely classes of the named entity. Classes of named entities can be, for instance, "Book", "Movie", "Game", and "Music". Given query "harry potter walkthrough", we detect "harry potter" as a named entity and assign "Game" to it as the most likely class, "Movie" and "Book" as less likely classes, and "Music" as unlikely class. This is because the context "walkthrough" strongly indicates that "harry potter" here is more likely to mean the Harry Potter game. (If the query is only "harry potter", then "Book" and "Movie" will be more plausible.)

NERQ is essentially useful for many applications in web search. According to our analysis, about 71% of search queries contain named entities. Identifying named entities in queries would help us to understand search intents better, and therefore provide better search. For example, in relevance search, we can improve ranking by treating named entity and context separately; in query suggestion, we can generate more relevant suggestions, e.g. "harry potter walkthrough" → "harry potter cheats" (context in the same class) or "halo 3 walkthrough" (entity in the same class).

As far as we know, there was no previous work on NERQ. Traditionally Named Entity Recognition (NER) is mainly performed on natural language texts [6, 3, 8]. Usually a supervised learning approach is exploited and a set of features (e.g., whether "Mr." occurs before the word, or whether the first letter of words is capitalized) is utilized. However, direct application of exiting NER technologies to NERQ would not perform well. This is because queries are usually very short (i.e., 2-3 words on average) and are not necessarily in standard form (e.g., all letters are in lower case), and thus the features are not sufficient for performing accurate NERQ.

In this paper, we propose a new probabilistic approach to NERQ using query log data. Without loss of generality, a query having one named entity[1] is represented as a triple (e, t, c), where e denotes named entity, t context of e, and c class of e. Note that t can be empty (i.e. no context), e.g. "harry potter". Then the goal of NERQ here becomes to find the triple (e, t, c) for a given query q, which has the largest joint probability $\Pr(e, t, c)$. The joint probability is factorized and then estimated by using query log and LDA.

In the LDA model, contexts of a named entity are represented as words of a document, classes of the named entity are represented as topics of the model. The alignment between model topics and predefined classes needs to be guaranteed. To address this problem, we propose a weakly supervised learning method, referred to as WS-LDA (Weakly Supervised Latent Dirichlet Allocation), which can leverage the weak supervision from humans.

[1]Here we only consider queries which contain single named entities. When there are multiple entities appearing in a query, our method can still be applied by viewing the more popular one as "named entity" and the rest as "context".

SIGIR'09, July 19–23, 2009, Boston, Massachusetts, USA.

Our approach is in part inspired by the work [19]. They proposed a method for acquiring named entities from query log using templates. There are some differences between their work and ours. Our focus is NERQ while theirs is offline query log mining (there is no online prediction in their case). We employ a probabilistic model, while they take a deterministic approach in the sense that they assume that each named entity can only belong to one class.

Our contribution in this paper lies in the following points. (1) We have formalized the problem of NERQ. (2) We have proposed a novel method for conducting NERQ. (3) We have developed a new topic modeling method with weakly supervised learning, i.e. WS-LDA.

The rest of the paper is organized as follows. Section 2 introduces related work. Section 3 defines the problem of NERQ and proposes a probabilistic approach to the task. Section 4 describes WS-LDA in details. Experimental results are presented in Section 5. Conclusions are made in the last section.

2. RELATED WORK

Needless to say, query processing is critically important for web search. Previous work mainly focused on query segmentation, query parsing, query classification, and query log mining. As far as we know, however, there was no work on Named Entity Recognition in Query (NERQ) as defined in this paper.

Query segmentation separates a query into a number of units [20, 2, 25]. However, it does not identify named entities from units and also does not assign class labels to units. Syntactic parsing focuses on identifying linguistic structure of query [9, 12, 13]. Query classification falls into two groups: (1) classification according to search intent, such as informational, navigational or transactional [7, 21, 16]; (2) classification according to semantics of query, such as "Shopping" or "Living" [24, 1]. In query classification, the whole query is classified and there is no further analysis on the internal structure of query.

Query log mining is also related to our work, particularly that by Paşca [19, 18, 23]. Paşca proposes a method for acquiring named entities in a class from query log. A query is supposed to consist of an instance (named entity) and a template (context). A bootstrapping method is employed to mine instances of a class by utilizing the templates of the class, starting with a small number of seed instances. Their approach is deterministic and it can only work well in the cases in which a named entity belongs to a single class.

Named Entity Recognition is usually performed on text documents. Early work on NER was based on rules [11]. Recently machine learning techniques have been applied to NER, including supervised machine learning [6, 3], semi-supervised learning [8] and unsupervised learning [10]. Features are utilized in these approaches. However, directly applying previous NER approaches to NERQ would not work well, because queries are usually short and not well formed.

Related work also includes topic modeling. Many topic models have been proposed including PLSI [15], LDA [5], and their extensions [14, 4]. Topic models have been utilized in topic discovery, document classification, citation analysis, and social network analysis. Our work exploits topic modeling in a new application, and is particularly unique in that it trains LDA with a weekly supervised learning method. There are several methods proposed for performing supervised learning of topic models [26, 17, 22]. In WS-LDA, we include weak supervision information as soft constraints in the objective function.

3. OUR APPROACH TO NERQ

3.1 NERQ Problem

Named Entity Recognition in Query (NERQ) is a task defined as follows. Given a query, we try to detect the named entities within query and categorize the named entities into classes. The classes are from a predefined taxonomy.

We have conducted a manual analysis on 1,000 *unique* queries randomly selected from the search log of a commercial web search engine. It indicates that named entities appear very frequently in queries and about 70% of the queries contain named entities. Furthermore, if a named entity occurs in a query, usually only that single named entity occurs and less than 1% of the queries contain two or more named entities. (In this paper, we focus on single-named-entity queries and take the processing of multiple named-entity queries as future work).

Queries tend to be short (i.e., 2-3 words on average) and not well formed. It makes NERQ a challenging task. In this paper, we propose a probabilistic approach to the problem using query log data.

3.2 Probabilistic Approach

A single-named-entity query q can be represented as triples (e, t, c), where e denotes named entity, t denotes the context of e in q, and c denotes the class of e. Note that t is further expressed as $\alpha\#\beta$, where α and β denote the left and right contexts respectively and # denotes a placeholder for named entity. Either α or β can be empty (e.g. "# walkthrough", "lyrics to #"), or both can be empty (i.e. "#"). For example, for query "harry potter walkthrough" belonging to Game, the associated triple is ("harry potter", "# walkthrough", Game).

The goal of NERQ is to detect the named entity e in query q, and assign the most likely class label c to e. Therefore, it can be accomplished by finding the triple $(e, t, c)^*$ among all possible triples, satisfying:

$$\begin{aligned}(e,t,c)^* &= \arg\max_{(e,t,c)} \Pr(q,e,t,c) \\ &= \arg\max_{(e,t,c)} \Pr(q|e,t,c)\Pr(e,t,c) \\ &= \arg\max_{(e,t,c)\in G(q)} \Pr(e,t,c) \qquad (1)\end{aligned}$$

In Eqn. (1), conditional probability $\Pr(q|e,t,c)$ represents how likely query q is generated from triple (e,t,c). Note that given a triple, it will uniquely determine a query. Therefore, for fixed query q and triple (e,t,c), $\Pr(q|e,t,c)$ can only be one or zero. That is, there are only two possibilities: either (e,t,c) generates q or (e,t,c) does not generate q. For instance, query "harry potter walkthrough" can be generated by ("harry potter", "# walkthrough", $*$), but not ("halo 3", "# walkthrough", $*$). We define $G(q)$ as the set containing all possible triples that can generate query q (i.e., $\Pr(q|e,t,c)$ equals one). Thus, the triple having largest probability $(e,t,c)^*$ must be in $G(q)$.

Therefore, to conduct NERQ we only need to calculate the joint probability $\Pr(e,t,c)$ for each triple in $G(q)$, which can be further factorized as below:

$$\begin{aligned}\Pr(e,t,c) &= \Pr(e)\Pr(c|e)\Pr(t|e,c) \\ &= \Pr(e)\Pr(c|e)\Pr(t|c) \qquad (2)\end{aligned}$$

In Eqn. (2), we assume that $\Pr(t_i|c) = \Pr(t_i|c, e_i)$, that is, context only depends on class but not specific named entity. This assumption largely reduces the parameter space and thus makes the learning tractable. It is also a reasonable assumption in practice because classes usually share common contexts, e.g., "Music" takes "# lyrics" and "# mp3" as contexts. There are contexts specific to named entities. However, due to data sparseness, one can hardly accurately estimate the probabilities of them.

The problem then becomes how to estimate $\Pr(e)$, $\Pr(c|e)$ and $\Pr(t|c)$. The number of such probabilities is extremely large, because there are an extremely large number of named entities and contexts. These include variants of named entities like "harry potter 6" and "harry potter and the half-blood prince", and variants of contexts like "# lyrics", "lyrics to #", and even typos "# lyrix".

3.3 Topic Model for NERQ

Suppose there is a training data set available, which contains triples from labeled queries $T = \{(e_i, t_i, c_i)|i = 1, \ldots, N\}$, where (e_i, t_i, c_i) denotes the "true" triple for query q_i and N is the data size. Therefore, the learning problem can be formalized as:

$$\max \prod_{i=1}^{N} \Pr(e_i, t_i, c_i) \tag{3}$$

If each named entity only belongs to one class, we can build the training data T easily (e.g., using the method in [19]). However, in reality named entities are usually ambiguous, e.g., "harry potter" can belong to classes "Book", "Movie", and "Game". It would be difficult as well as time-consuming to manually assign class labels to named entities in queries. Therefore, we collect training data $T = \{(e_i, t_i)\}$, and view class label c_i as hidden variable. We also know the possible classes of each named entity in training. The learning problem with respect to the new training data $T = \{(e_i, t_i)\}$ becomes:

$$\max \prod_{i=1}^{N} \Pr(e_i, t_i) = \max \prod_{i=1}^{N} \Pr(e_i) \sum_{c} \Pr(c|e_i) \Pr(t_i|c) \tag{4}$$

In Eqn. (4), $\Pr(e_i)$ represents the popularity of named entity e_i, $\Pr(c|e_i)$ represents the likelihood of class c given named entity e_i, and $\Pr(t_i|c)$ represents the likelihood of context t_i given class c. The prior probability $\Pr(e_i)$ can be estimated in different ways, independent of $\Pr(c|e_i)$ and $\Pr(t_i|c)$. Suppose it is estimated as $\hat{\Pr}(e_i)$, then Eqn. (4) becomes:

$$\max \prod_{i=1}^{N} \hat{\Pr}(e_i) \prod_{i=1}^{N} \sum_{c} \Pr(c|e_i) \Pr(t_i|c) \tag{5}$$

In this way, the learning problem becomes that of learning the probabilities in Eqn. (5), which form a topic model. In the topic model, a named entity corresponds to a document, contexts of a named entity correspond to words of the document, classes of a named entity correspond to topics of the model. Without loss of generality, we choose LDA as topic model in this paper. The topic model also has some specialties. The topics (or classes) in the topic model are predefined and the possible topics of each document are given in training.

3.4 Implementation

In this section, we explain how to use the topic model and query log to build a NERQ system. The processing consists of two stages, offline training and online prediction.

3.4.1 Offline Training

The offline training process is a combination of learning algorithm and data mining technique. There are two steps:

(1) We first select some named entities as seeds, and assign possible classes to each of them. There might be multiple classes for each named entity. Note that the effort in this labeling is limited, since we only need to label a small number of named entities (not queries). Then we scan the query log with the seed named entities and collect all the queries containing them. In this way, we can generate the training data (e_i, t_i) and learn a topic model with regard to the seed named entities. There is a significant difference between conventional topic modeling and the learning here. First, the hidden topics (or classes) are predefined. Furthermore, the possible topics (classes) of a document (named entity) are given in weak supervision. We propose a method that can conduct weakly supervised learning of topic model, referred to as WS-LDA (Weakly Supervised Latent Dirichlet Allocation). We will introduce the details in the next section. After this step, we obtain the estimated probabilities $\Pr(c|e)$ for each seed named entity as well as $\Pr(t|c)$ for each class.

(2) We scan the query log again with the previously learned contexts, collect all the queries containing the contexts, and extract the remainder of these queries as new named entities. (To ensure a high quality extraction, we heuristically make a threshold cut-off in this process). Next, WS-LDA is employed to estimate $\Pr(c|e)$ for the newly extracted named entities, with the probabilities $\Pr(t|c)$ fixed. The probabilities $\Pr(e)$ for newly extracted named entities are also estimated in this process. Specifically, we use the total frequency of queries containing e in the query log to approximate $\Pr(e)$. The more frequently named entity e occurs, the larger probability $\Pr(e)$ will be.

In this way, we can estimate all the probabilities we need, that is, $\Pr(e)$, $\Pr(c|e)$, and $\Pr(t|c)$. We create an index for the named entities and the classes, and store the estimated probabilities for efficient online prediction. The detailed algorithm for the offline training process is shown in Alg. 1.

3.4.2 Online Prediction

In online prediction, we try to find the most likely triples in $G(q)$ for a query q. We can generate $G(q)$ by segmenting the query into named entity and context in all possible ways, and labeling segmented named entities with all possible classes. For each triple (e, t, c) in $G(q)$, the joint probability $\Pr(e, t, c)$ is then calculated. The triples with highest probabilities are output results for NERQ. The detailed algorithm is shown in Alg. 2. The time complexity of the algorithm is $O(kn^2)$, where k denotes number of classes and n denotes number of words in a query. Since both k and n are very small, the prediction can be conducted very efficiently. We skip those queries which do not have named entity and context stored in the index.

4. WS-LDA

The learning of topic model in our method for NERQ is a new problem, since the topics in the model are predefined,

Algorithm 1 Offline Training Algorithm

Input: Repository of queries Q, set of classes C, set of seed named entities S with their labels $C_s, s \in S$
Output: Context set T, class index I_C, and named entity index I_E
Variable: T_* = context of named entity $*$,
E = pool of named entities

initialize $I_C \leftarrow \varnothing, I_E \leftarrow \varnothing$
$T \leftarrow \varnothing$
for all $q \in Q$ **do**
  for all $s \in S$ **do**
    if (q contains s) **then**
      $t \leftarrow RemainderContext(q, s)$
      $T = T \cup t$
      update t's information in T_s
    end if
  end for
end for
train topic model WS-LDA over $(S, \{T_s\}_{s\in S}, \{C_s\}_{s\in S})$
store learned probabilities $\{\Pr(t|c)\}_{t\in T, c\in C}$ into I_C
$E \leftarrow \varnothing$
for all $q \in Q$ **do**
  for all $t \in T$ **do**
    if (q contains t) **then**
      $e \leftarrow QueryRemainderEntity(q, t)$
      $E = E \cup e$
      update t's information in T_e
    end if
  end for
end for
cut off E to retain high quality named entities
for all $e \in E$ **do**
  estimate $\Pr(e)$ with T_e
  estimate $\{\Pr(c|e)\}_{c\in C}$ with T_e and learned WS-LDA
  store $(\Pr(e), \{\Pr(c|e)\}_{c\in C})$ in I_E
end for
return (T, I_C, I_E)

Table 1: Relationship between Notions

Query	Document	Symbol
Context	Word	w_n
Named entity	Document	$\boldsymbol{w}$
Class	Topic	z_n

and the possible topics of document are given. We propose a new method for learning topic model, WS-LDA (Weakly Supervised Latent Dirichlet Allocation). WS-LDA is a general method and can be used in other applications.

For readability, we use conventional notations for document processing to describe the topic model. Specifically, contexts become "words", contexts of a named entity form a "document", and classes of named entity correspond to "topics". Suppose that we have named entity "harry potter" with classes "Movie", "Book", and "Game", and find three queries containing "harry potter" in the query log, "harry potter movie", "harry potter walkthrough", and "harry potter review". Then the document with respect to "harry potter" will contain three words, i.e. "# movie", "# walkthrough", and "# review", and the topics of the document will be "Movie", "Book", and "Game". The relationship between query data and document data is summarized in Table 1.

Accordingly, we can rewrite the topic model in Eqn. (5) in the following form for better understanding:

$$\prod_{e} \prod_{\{i|e=e_i\}} \sum_{c} \Pr(c|e_i)\Pr(t_i|c) \tag{6}$$

where e denotes a unique named entity in training data. Please note that $\hat{\Pr}(e_i)$ is dropped for clarity, and it can be easily integrated into the model. The first product in Eqn. (6) is on all the unique named entities in the training data (document level product), and the second one is on all the contexts of the same named entity (word level product).

Algorithm 2 Online Prediction Algorithm

Input: Query $q = w_1 w_2 \dots w_n$, a set of classes C, context set T, named entity index I_E and class index I_C
Output: Top K recognition results R

initialize $R \leftarrow \varnothing$
for $i = 1$ to n **do**
  for $j = i$ to n **do**
    $e \leftarrow w_i w_{i+1} \dots w_j$
    $t \leftarrow w_1 w_2 \dots w_{i-1} \# w_{j+1} w_{j+2} \dots w_n$
    if ($e \in I_E$ and $t \in T$) **then**
      for all $c \in C$ **do**
        $r \leftarrow$ new recognition
        $r.triple \leftarrow \{e, t, c\}$
        compute $\Pr(e), \Pr(c|e), \Pr(t|c)$ using I_E and I_C
        $r.prob \leftarrow \Pr(e)\Pr(c|e)\Pr(t|c)$
        $R.push(r)$
      end for
    end if
  end for
end for
sort R by $prob$
truncate R to size K
return R

4.1 Model

We first give the definition of the model in WS-LDA, which is the same as the conventional LDA. Suppose there is a corpus of M documents $\mathcal{D} = \{\boldsymbol{w}_1, ..., \boldsymbol{w}_M\}$ sharing K topics, and each document is a sequence of N words denoted by $\boldsymbol{w} = \{w_1, ..., w_N\}$. It is assumed that the documents $\boldsymbol{w}$ in the corpus $\mathcal{D}$ are generated by the following generative process:

1. Draw topic distribution $\boldsymbol{\theta} \sim$ Dirichlet($\boldsymbol{\alpha}$)
2. For each word
   (a) Draw topic assignment $z_n \sim$ Multinomial($\boldsymbol{\theta}$)
   (b) Draw word $w_n \sim$ Multinomial($\boldsymbol{\beta}_{z_n}$), a multinomial distribution conditioned on topic z_n

Given parameters $\Theta = \{\boldsymbol{\alpha}, \boldsymbol{\beta}\}$, we obtain the probability distribution of a document:

$$p(\boldsymbol{w}|\Theta) = \int p(\boldsymbol{\theta}|\boldsymbol{\alpha})(\prod_{n=1}^{N} \sum_{z_n} p(z_n|\boldsymbol{\theta})p(w_n|z_n, \boldsymbol{\beta}))d\boldsymbol{\theta} \tag{7}$$

Finally, taking the product of probabilities of documents, we obtain the probability of corpus:

$$p(\mathcal{D}|\Theta) = \prod_{d=1}^{M} \int p(\boldsymbol{\theta_d}|\boldsymbol{\alpha})(\prod_{n=1}^{N_d} \sum_{z_{dn}} p(z_{dn}|\boldsymbol{\theta_d})p(w_{dn}|z_{dn}, \boldsymbol{\beta}))d\boldsymbol{\theta_d}$$

4.2 Weak Supervision

In NERQ, employing an unsupervised learning method to learn the topic model would not work. This is because the topics (classes) are explicitly predefined in NERQ. In contrast, the topics in a conventional topic model are implicit and are automatically learned. There is no guarantee that the hidden topics learned by unsupervised learning method will be aligned with the predefined topics (classes). Therefore, we need to introduce supervision in the training process of the topic model.

The supervision is from the manual class labels on each seed named entity. The labels are not exclusive because ambiguity exists in named entities. For example, "harry potter" may have three classes, i.e. "Movie", "Book", and "Game". We only ask human judges to make a judgment on whether a named entity can belong to a class or not. (It would be extremely hard for human judges to decide a

probability of a named entity's belonging to a class.) This type of labels is viewed as *weak* supervision for training. That means, in the terminology of topic modeling, we only assume that a document has high probabilities on *labeled topics*, but very low probabilities on *unlabeled topics*.

4.2.1 Objective Function

Given document $\boldsymbol{w}$, the assigned class labels are represented as $\boldsymbol{y} = \{y_1, ..., y_K\}$, where y_i takes 1 or 0 when the i-th topic is or is not assigned to the document, and K denotes the number of topics. The weak supervision information will be used as soft constraints in the objective function. WS-LDA tries to maximize the likelihood of data with respect to the model, and at the same time satisfy the soft constraints. The constraints are defined as follows.

$$\mathcal{C}(\boldsymbol{y}, \Theta) = \sum_{i=1}^{K} y_i \bar{z}_i \tag{8}$$

Here we let $\bar{z}_i = \frac{1}{N}\sum_{n=1}^{N} z_n^i$, where z_n^i is 1 or 0 when the i-th topic is or is not assigned to the n-th word. That is to say, $\bar{z}_i$ represents the empirical probability of the i-th topic in document $\boldsymbol{w}$. As we can see, maximizing the soft constraints actually can meet the following two goals at the same time: (1) the i-th latent topic is aligned to the i-th predefined class; and (2) the document $\boldsymbol{w}$ is mainly distributed over labeled classes.

Specifically, the objective function with respect to a document is defined as follows.

$$\mathcal{O}(\boldsymbol{w}|\boldsymbol{y}, \Theta) = \log p(\boldsymbol{w}|\Theta) + \lambda \mathcal{C}(\boldsymbol{y}, \Theta) \tag{9}$$

where likelihood function $p(\boldsymbol{w}|\Theta)$ and soft constraint function $\mathcal{C}(\boldsymbol{y}, \Theta)$ are represented as in Eqn. (7) and (8) respectively, and λ is coefficient. If λ equals 0, WS-LDA learning will degenerate to LDA learning.

Finally, substituting Eqn. (7) and (8) into Eqn. (9) and taking the sum over all documents, we obtain the following total objective function:

$$\begin{aligned}\mathcal{O}(\mathcal{D}|\mathcal{Y}, \Theta) &= \sum_{d=1}^{M} \mathcal{O}(\boldsymbol{w_d}|\boldsymbol{y_d}, \Theta) \\ &= \sum_{d=1}^{M} \log \int p(\boldsymbol{\theta_d}|\boldsymbol{\alpha}) (\prod_{n=1}^{N_d} \sum_{z_{dn}} p(z_{dn}|\boldsymbol{\theta_d}) p(w_{dn}|z_{dn}, \boldsymbol{\beta})) d\boldsymbol{\theta_d} \\ &\quad + \sum_{d=1}^{M} \lambda \sum_{i=1}^{K} y_{di} \bar{z}_{di}\end{aligned} \tag{10}$$

4.2.2 Algorithm

WS-LDA is equivalent to maximizing the objective function in Eqn. (10). However, there might be no analytic solution for the problem as in conventional LDA learning. Therefore, we employ a variational method similar to that in [5] to approximate the posterior distribution of the latent variables. The approximate distribution is characterized by the following variational distribution:

$$q(\boldsymbol{\theta}, \boldsymbol{z}|\Lambda) = q(\boldsymbol{\theta}|\boldsymbol{\gamma}) \prod_{n=1}^{N} q(z_n|\phi_n)$$

where $\Lambda = \{\boldsymbol{\gamma}, \phi_{1:N}\}$ are variational parameters. Specifically, $\boldsymbol{\gamma}$ is Dirichlet parameter and $\phi_{1:N}$ are multi-nominal parameters.

Therefore, the objective function for a single document can be derived as follows.

$$\mathcal{O}(\boldsymbol{w}|\boldsymbol{y}, \Theta) = L(\Lambda; \Theta) + D(q(\boldsymbol{\theta}, \boldsymbol{z}|\Lambda) || p(\boldsymbol{\theta}, \boldsymbol{z}|\boldsymbol{w}, \Theta)) \tag{11}$$

where

$$\begin{aligned}L(\Lambda; \Theta) &= \int \sum_{\boldsymbol{z}} q(\boldsymbol{\theta}, \boldsymbol{z}|\Lambda) \log \frac{p(\boldsymbol{\theta}, \boldsymbol{z}, \boldsymbol{w}|\Theta)}{q(\boldsymbol{\theta}, \boldsymbol{z}|\Lambda)} d\boldsymbol{\theta} \\ &+ \int \sum_{\boldsymbol{z}} q(\boldsymbol{\theta}, \boldsymbol{z}|\Lambda) \lambda \mathcal{C}(\boldsymbol{y}, \Theta) d\boldsymbol{\theta}\end{aligned}$$

Minimizing the KL divergence between the variational posterior probability and the true posterior probability, denoted as $D(q(\boldsymbol{\theta}, \boldsymbol{z}|\Lambda) || p(\boldsymbol{\theta}, \boldsymbol{z}|\boldsymbol{w}, \Theta))$, gives a good approximate distribution of $p(\boldsymbol{\theta}, \boldsymbol{z}|\boldsymbol{w}, \Theta)$. From Eqn. (11) we can see, this is equivalent to maximizing the lower bound $L(\Lambda; \Theta)$ on the objective function $\mathcal{O}(\boldsymbol{w}|\boldsymbol{y}, \Theta)$ with respect to Λ which has the form

$$\begin{aligned}\mathcal{O}(\boldsymbol{w}|\boldsymbol{y}, \Theta) &\geq L(\Lambda; \Theta) \\ &= E_q[\log p(\boldsymbol{\theta}|\boldsymbol{\alpha})] + E_q[\log p(\boldsymbol{z}|\boldsymbol{\theta})] + E_q[\log p(\boldsymbol{w}|\boldsymbol{z}, \boldsymbol{\beta})] \\ &\quad - E_q[\log q(\boldsymbol{\theta})] - E_q[\log q(\boldsymbol{z})] + E_q[\lambda \mathcal{C}(\boldsymbol{y}, \Theta)]\end{aligned}$$

Let β_{iv} be $p(w_n^v = 1|z^i = 1)$ for word v. Each of the above terms can be expressed in the following equations (12)$\sim$(17):

$$\begin{aligned}L(\Lambda; \Theta) &= \log \Gamma(\textstyle\sum_{j=1}^{K} \alpha_j) - \sum_{i=1}^{K} \log \Gamma(\alpha_i) \\ &\quad + \textstyle\sum_{i=1}^{K} (\alpha_i - 1)(\Psi(\gamma_i) - \Psi(\sum_{j=1}^{K} \gamma_j)) && (12) \\ &\quad + \textstyle\sum_{n=1}^{N} \sum_{i=1}^{K} \phi_{ni} (\Psi(\gamma_i) - \Psi(\sum_{j=1}^{K} \gamma_j)) && (13) \\ &\quad + \textstyle\sum_{n=1}^{N} \sum_{i=1}^{K} \sum_{v=1}^{V} \phi_{ni} w_n^v \log \beta_{iv} && (14) \\ &\quad - \log \Gamma(\textstyle\sum_{j=1}^{K} \gamma_j) + \sum_{i=1}^{K} \log \Gamma(\gamma_i) \\ &\quad - \textstyle\sum_{i=1}^{K} (\gamma_i - 1)(\Psi(\gamma_i) - \Psi(\sum_{j=1}^{K} \gamma_j)) && (15) \\ &\quad - \textstyle\sum_{n=1}^{N} \sum_{i=1}^{K} \phi_{ni} \log \phi_{ni} && (16) \\ &\quad + \frac{\lambda}{N} \textstyle\sum_{n=1}^{N} \sum_{i=1}^{K} y_i \phi_{ni} && (17)\end{aligned}$$

Notice that

$$E_q[\bar{z}_i] = E_q[\frac{1}{N}\sum_{n=1}^{N} z_n^i] = \frac{1}{N}\sum_{n=1}^{N} E_q[z_n^i] = \frac{1}{N}\sum_{n=1}^{N} \phi_{ni}$$

is used for the derivation of the term (17).

A variational expectation-maximization (EM) algorithm is then employed to estimate the model parameters Θ.

E-step:

$$\gamma_i = \alpha_i + \textstyle\sum_{n=1}^{N} \phi_{ni}$$

$$\phi_{ni} \propto \beta_{iv} \exp(\Psi(\gamma_i) - \Psi(\textstyle\sum_{j=1}^{K} \gamma_j) + \frac{\lambda}{N} y_i)$$

M-step:

$$\beta_{ij} \propto \textstyle\sum_{d=1}^{M} \sum_{n=1}^{N_d} \phi_{dni} w_{dn}^j$$

Dirichlet parameter $\boldsymbol{\alpha}$ can be updated in the M-step by using an efficient Newton-Raphson method in which the inverted Hessian can be computed in linear time.

4.3 Prediction

WS-LDA is also used in prediction. Specifically, we calculate the probability $\Pr(c|e)$ for unseen named entities in NERQ. This corresponds to estimating the probability of topic given a new document $\boldsymbol{w}$ with the already estimated model Θ. The estimation is then equivalent to approximating the posterior topic distribution $\boldsymbol{\theta}$ of the new document $\boldsymbol{w}$ using the variational inference procedure. Notice this is the same as variational inference in conventional LDA (cf., [5]).

5. EXPERIMENTAL RESULTS

We conducted experiments to verify the effectiveness of NERQ using WS-LDA. In this section, we first introduce the data sets used in experiments. Then we demonstrate the effectiveness of our approach in NERQ. Finally, we compare our method of NEQR using WS-LDA with two baseline methods, the deterministic approach proposed in [19], referred to as Determ, and conventional LDA (unsupervised learning), referred to as LDA. Note that although LDA is viewed as a baseline, there was no previous work on using LDA in NERQ. In the experiments λ was set to 1 by default.

5.1 Data Set

We made use of a real data set consisting of over *6 billion* queries, in which the number of unique queries is *930 million*. The queries were randomly sampled from the query log of a commercial web search engine.

Four semantic classes were considered in our experiments, including "Movie", "Game", "Book", and "Music". Based on these classes, 180 named entities were selected from the web sites of Amazon, GameSpot, and Lyrics. Four human annotators labeled the classes of the named entities. If there was a disagreement among the annotators, we took a majority voting. Multiple classes can be assigned to one named entity. The annotated data was further divided into a training set containing 120 named entities and a test set containing 60 named entities.

The data set has the following characteristics. First, the overlap ratios between classes vary according to class pairs, e.g. the "Movie" and "Game" classes as well as the "Movie" and "Book" classes have higher overlap ratios ($\geq$ 20%). It seems natural because a movie is often adapted from a book with the same title, or a game is often inspired by a movie and named after the movie. Second, the selected classes differ from one another in terms of frequency in query log, e.g. named entities in "Movie" and "Game" classes occur more frequently than in "Book" and "Music" classes.

Starting from the 120 seed named entities, we trained a WS-LDA model for conducting NERQ. Specifically, we extracted all the possible contexts of seed named entities, and created a WS-LDA model as described in Sections 3 and 4. Finally we obtained 432,304 contexts and indexed about 1.5 million named entities.

5.2 NERQ by WS-LDA

We conducted NERQ on queries from a separate query log, which consists of about 12 million unique queries, and obtained about 0.14 million recognition results. We randomly sampled 400 queries from the recognition results for evaluation. Table 2 gives some examples from the data set and Table 3 shows the number of queries in the data set grouped by the predicted classes of named entities.

Each recognition result was then manually labeled as "correct" or "incorrect". A result is viewed as correct *if and only if both the detection and classification of the named entity are correct*. The performance of NERQ is evaluated in terms of top N accuracy. "Top N accuracy" here is defined in the following way: an algorithm output will be considered "correct" if at least one of top N results is labeled as "correct".

Fig. 1 shows the accuracy of our NERQ method in terms of top N accuracy. "Overall" stands for the average performance of NERQ over all classes. From Fig. 1 we can see that the overall top 1 accuracy is 81.75% which is reasonably good. When we consider the top 3 results, we can even make the overall accuracy reach 97.5%. Fig. 1 also shows the performances of NERQ in different classes. From the results we can see that our method of NERQ using WS-LDA is effective in each class.

Table 2: Example Queries

pics of fight club	braveheart quote
watch gladiator online	american beauty company
12 angry men characters	mario kart guide
pc mass effect	crysis mods
mother teresa images	condemned screenshots
4 minutes lyric	king kong
the black swan summary	blackwater novel
new moon	rehab the song
nineteen minutes synopsis	umbrella chords
all summer long video	girlfriend lyrics

Table 3: Statistics on Sampled Recognition Results

	Movie	Game	Book	Music
Num. of queries	111	108	82	99

We further made error analysis on our NERQ results. There were mainly three types of errors. (1) Errors were mainly caused by inaccurate estimation of $\Pr(e)$. It seems that the current way of estimating $\Pr(e)$ has certain bias, which prefers the segmentation with a shorter named entity. We may reduce such kind of errors by employing a better estimation method. (2) Some contexts were not learned in our approach since they are uncommon. For example, in the query "lyrics for forever by chris brown", "forever by chris brown" was recognized as a "Music" named entity and "lyrics for #" the context. Ideally, "forever" should be recognized as named entity of "Music", and "lyrics for # by chris brown" as context. However, since the context "lyrics for # by chris brown" is quite specific, it was not covered by our learning method. Some of such errors may be eliminated by using more seed named entities. (3) Some queries contained the named entity out of predefined classes. For example, in query "american beauty company", "american beauty" was incorrectly recognized as a movie name. Since "american beauty" was indexed as a movie name and "# company" was as a common context, our NERQ system may occasionally make such kind of errors. We may reduce them when we utilize more classes.

5.3 WS-LDA v.s. Baselines

We performed experiments to make comparison between the WS-LDA approach and two baseline methods: Determ and LDA. Note that the main difference of these approaches lies in different assumptions and ways for modeling the relationship between named entity, context, and class.

Determ learns the contexts of a certain class by simply aggregating all the contexts of named entities belonging to that class. It can perform very well when a named entity only belongs to a single class. In contrast, LDA and WS-LDA take a probabilistic approach and handle the ambiguity of named entities. However, LDA is based on unsupervised learning, and thus cannot ensure the alignment between latent classes and predefined classes.

5.3.1 Modeling Contexts of Class

We first compared the learning of contexts of each class between WS-LDA and two baselines. Table 4 shows the top ranked contexts of each class according to $\Pr(t|c)$ generated

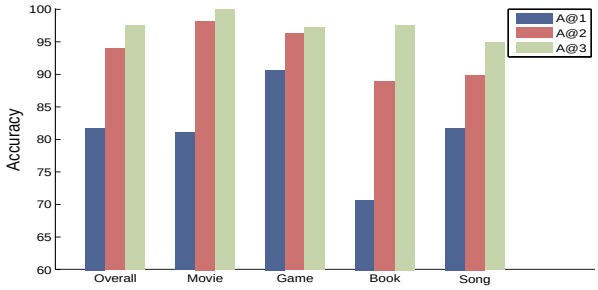

Figure 1: NERQ Top N Accuracy on Test Data(%)

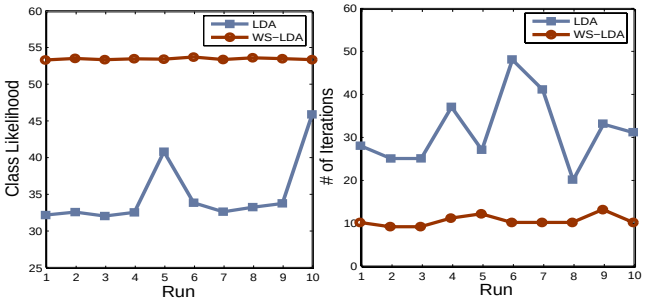

Figure 2: Comparisons between WS-LDA and LDA on (a) Overall Class Likelihood on Testing Set, (b) Convergence Speed on Training Set

by WS-LDA approach and baselines. From the results we can see that the quality of the top ranked contexts generated by Determ is not high. Take the "Movie" class as example, its top ranked contexts are mixed with the contexts of the "Game" class, e.g. "# games" or "free online # games". The reason is that there are many named entities belonging to both "Movie" and "Game" classes. However, Determ ignores the ambiguity and forcibly merges all the contexts of such named entities together. The results indicate that by taking a probabilistic approach, we can improve the quality of the learned contexts. Among the two topic model approaches, WS-LDA achieves better results than LDA, because it can leverage human supervision. Note here manual class alignment is performed to make it a fair comparison with LDA.

We further looked at the accuracy in ranking named entities by WS-LDA and Determ. In Determ, all the contexts of a class are called the signature of the class. Candidate named entities can then be ranked in each class based on the Jensen-Shannon similarity score [19] between all the contexts of the named entity and the signature of the class. For comparison, top 250 named entities ranked in each class generated by Determ and WS-LDA were evaluated. Each named entity was manually labeled as "correct" or "incorrect" regarding to class. In total, 2,000 named entities for the four predefined classes were annotated. We used "precision at rank N" [19] as the measure and obtained the results in Table 5. The results show that WS-LDA can significantly outperform Determ (p-value<0.01). The results demonstrate that WS-LDA can learn the contexts of classes better than Determ.

5.3.2 *Class Prediction and Convergence Speed*

We next evaluated the accuracies of estimated probabilities $\Pr(c|e)$ in LDA and WS-LDA on the test data (i.e., 60 named entities). The overall class likelihood with respect to named entity e is calculated as $\sum_{i=1}^{K} y_i \Pr(c_i|e)$, where y_i takes 1 or 0 when the i-th class is or is not assigned to e. The overall class likelihood measures how consistent the machine predictions are with human labels. Fig. 2(a) shows the results by LDA and WS-LDA over different runs in testing. The results indicate that WS-LDA significantly outperforms LDA. The average likelihood obtained by LDA is about 34.89, while the average likelihood obtained by WS-LDA is about 53.39. Here to avoid inaccurate manual class alignment in LDA, we enumerate $K!$ possible alignments for LDA in each run and take the highest score as its result. It can be considered as the upper-bound of LDA. As shown in Fig. 2(a), the performance of LDA is quite unstable. It might be also related to the "local maximum" problem of LDA [5]. In contrast, WS-LDA model does not seem to suffer from the problem and can constantly produce high accuracy in prediction.

Table 6: Average Class Likelihood on Testing Set v.s. Coefficient λ

Coefficient λ	0.01	0.1	1	10	100
ACL	20.92	35.92	53.39	53.22	53.08

We also found that the convergence speed of training for WS-LDA is much faster than that for LDA. Fig. 2(b) shows numbers of iterations needed for convergence for the two methods. The average convergence speed of WS-LDA is 3 times faster than that of LDA. It might be due to the regulation from the soft constraint which makes the parameter space much smaller.

5.4 Supervision in WS-LDA

We also tested how the coefficient λ (weight on soft constraints) affects the performance of WS-LDA. We set λ with different values from 0.01 to 100 and ran 10 trials under each setting. Table 6 shows the average class likelihood values on the testing set under different values of λ. The results indicate that increasing λ will help WS-LDA to predict class labels more accurately. It demonstrates the necessity of using the supervision in learning and the capability of WS-LDA in utilizing the information. While λ is small, the average class likelihood is unsurprisingly close to that of LDA. Moreover, when λ continues to increase, the convergence speed will decrease and the performance will drop. This is because the supervision is over emphasized.

6. CONCLUSIONS

Named Entity Recognition in Query (NERQ) is potentially useful in many applications in web search. We have, for the first time, investigated the problem in this paper, and proposed employing a probabilistic approach to perform the task using query log and a topic model. We have proposed a new weakly supervised learning method for creating the topic model called WS-LDA, in which the topics of a document are assigned. Experimental results indicate that the proposed approach can accurately perform NERQ, and outperforms other baseline methods.

There are several issues which we plan to address in the future. In this paper, we have verified the effectiveness of our method in experiments in which there are only a small number of classes. We plan to add more classes and conduct the experiments. The proposed method focuses on single-named-entity queries. We want to design a more general model to handle more complicated queries.

Table 4: Comparisons on Learned Contexts of Each Class

Movie			Game		
Determ	LDA	WS-LDA	Determ	LDA	WS-LDA
lego #	# movie	# movie	# cheats	# games	# games
# games	# wallpaper	# photos	# movie	# cheats	# cheats
# wallpaper	# movies	# soundtrack	# games	# wallpaper	lego #
# characters	lego #	# pics	# cheat codes	lego #	# download
# toys	# games	# movies	# the movie	# download	# wallpaper
# movie	# cast	# the movie	# wallpaper	# online	play # online
# pictures	# imagesize large	# wallpaper	# download	play # online	# online
free online # games	# game	# cast	play # online	# cheat codes	# cheat codes
free # games	# video	# pictures	# game	# game	# game
new # movie	# trilogy	# imagesize large	# logo	new # movie	download #
Book			**Music**		
Determ	LDA	WS-LDA	Determ	LDA	WS-LDA
# movie	# summary	# summary	# lyrics	# lyrics	# lyrics
# soundtrack	# book	# book	# movie	# film	# video
# book	# review	# review	# photos	# video	# song
movie #	# film	# synopsis	# film	# song	lyrics #
# quotes	# synopsis	summary of #	# song	# star	lyrics to #
# imagesize large	# star	book #	# pics	# director	lyrics for #
# dvd	# director	# quotes	# soundtrack	lyrics #	# song lyrics
# review	summary of #	# reviews	# video	lyrics to #	# l
# trailer	book #	# video	# star	lyrics for #	# quotes
# the book	# quotes	# author	# cast	# song lyrics	lyrics of #

Table 5: Comparisons on Learned Named Entities of Each Class (P@N)

	Movie		Game		Book		Music		Average-Class	
	Determ	WS-LDA	Determ	WS-LDA	Determ	WS-LDA	Determ	WS-LDA	Determ	WS-LDA
P@25	0.92	1	0.98	1	0.84	1	0.96	1	0.92	1
P@50	0.9	1	0.96	1	0.82	1	0.92	1	0.905	1
P@100	0.85	1	0.93	0.98	0.79	0.98	0.89	1	0.865	0.99
P@150	0.82	1	0.92	0.953	0.767	0.98	0.833	1	0.835	0.983
P@250	0.724	0.988	0.896	0.928	0.732	0.968	0.76	0.984	0.778	0.967

7. REFERENCES

[1] S. M. Beitzel, E. C. Jensen, O. Frieder, D. Grossman, D. D. Lewis, A. Chowdhury, and A. Kolcz. Automatic web query classification using labeled and unlabeled training data. In *SIGIR '05*, pages 581–582, 2005.

[2] S. Bergsma and Q. I. Wang. Learning noun phrase query segmentation. In *EMNLP-CoNLL '07*, pages 819–826, 2007.

[3] D. M. Bikel, S. Miller, R. Schwartz, and R. Weischedel. Nymble: a high-performance learning name-finder. In *ANLC '97*, pages 194–201, 1997.

[4] D. M. Blei and J. D. Lafferty. Dynamic topic models. In *ICML '06*, pages 113–120, 2006.

[5] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, 2003.

[6] A. E. Borthwick. *A maximum entropy approach to named entity recognition*. PhD thesis, New York, NY, USA, 1999.

[7] A. Broder. A taxonomy of web search. *SIGIR Forum*, 36:3–10, 2002.

[8] M. Collins and Y. Singer. Unsupervised models for named entity classification. In *In Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 100–110, 1999.

[9] E. F. de Lima and J. O. Pedersen. Phrase recognition and expansion for short, precision-biased queries based on a query log. In *SIGIR '99*, pages 145–152, 1999.

[10] O. Etzioni, M. Cafarella, D. Downey, A.-M. Popescu, T. Shaked, S. Soderland, D. S. Weld, and A. Yates. Unsupervised named-entity extraction from the web: an experimental study. *Artif. Intell.*, 165:91–134, June 2005.

[11] R. Gaizauskas, K. Humphreys, H. Cunningham, and Y. Wilks. University of Sheffield: description of the LaSIE system as used for MUC-6. In *MUC6*, pages 207–220, 1995.

[12] J. Gao and J.-Y. Nie. A study of statistical models for query translation: finding a good unit of translation. In *SIGIR '06*, pages 194–201, 2006.

[13] J. Gao, J.-Y. Nie, and M. Zhou. Statistical query translation models for cross-language information retrieval. *ACM TALIP*, 5(4):323–359, 2006.

[14] T. L. Griffiths, M. Steyvers, D. M. Blei, and J. B. Tenenbaum. Integrating topics and syntax. In *NIPS*, pages 537–544. MIT Press, 2005.

[15] T. Hofmann. Probabilistic latent semantic indexing. In *SIGIR '99*, pages 50–57, 1999.

[16] U. Lee, Z. Liu, and J. Cho. Automatic identification of user goals in web search. In *WWW '05*, pages 391–400, 2005.

[17] A. K. Mccallum. Multi-label text classification with a mixture model trained by EM. In *In AAAI 99 Workshop on Text Learning*, 1999.

[18] M. Paşca. Organizing and searching the world wide web of facts – step two: harnessing the wisdom of the crowds. In *WWW '07*, pages 101–110, 2007.

[19] M. Paşca. Weakly-supervised discovery of named entities using web search queries. In *CIKM '07*, pages 683–690, 2007.

[20] K. M. Risvik, T. Mikolajewski, and P. Boros. Query segmentation for web search. In *WWW*, 2003.

[21] D. E. Rose and D. Levinson. Understanding user goals in web search. In *WWW '04*, pages 13–19, 2004.

[22] I. Sato and H. Nakagawa. Knowledge discovery of multiple-topic document using parametric mixture model with Dirichlet prior. In *KDD '07*, pages 590–598, 2007.

[23] S. Sekine and H. Suzuki. Acquiring ontological knowledge from query logs. In *WWW '07*, pages 1223–1224, 2007.

[24] D. Shen, J.-T. Sun, Q. Yang, and Z. Chen. Building bridges for web query classification. In *SIGIR '06*, pages 131–138, 2006.

[25] B. Tan and F. Peng. Unsupervised query segmentation using generative language models and Wikipedia. In *WWW '08*, pages 347–356. ACM, 2008.

[26] N. Ueda and K. Saito. Parametric mixture models for multi-labeled text. In *NIPS*, pages 721–728. MIT Press, Cambridge, MA, 2003.