Clickthrough-Based Latent Semantic Models for Web Search

Jianfeng Gao
Microsoft Research
One Microsoft Way
Redmond, WA 98052 USA
jfgao@microsoft.com

Kristina Toutanova
Microsoft Research
One Microsoft Way
Redmond, WA 98052 USA
kristout@microsoft.com

Wen-tau Yih
Microsoft Research
One Microsoft Way
Redmond, WA 98052 USA
scottyih@microsoft.com

ABSTRACT

This paper presents two new document ranking models for Web search based upon the methods of semantic representation and the statistical translation-based approach to information retrieval (IR). Assuming that a query is parallel to the titles of the documents clicked on for that query, large amounts of query-title pairs are constructed from clickthrough data; two latent semantic models are learned from this data. One is a bilingual topic model within the language modeling framework. It ranks documents for a query by the likelihood of the query being a semantics-based translation of the documents. The semantic representation is language independent and learned from query-title pairs, with the assumption that a query and its paired titles share the same distribution over semantic topics. The other is a discriminative projection model within the vector space modeling framework. Unlike Latent Semantic Analysis and its variants, the projection matrix in our model, which is used to map from term vectors into sematic space, is learned discriminatively such that the distance between a query and its paired title, both represented as vectors in the projected semantic space, is smaller than that between the query and the titles of other documents which have no clicks for that query. These models are evaluated on the Web search task using a real world data set. Results show that they significantly outperform their corresponding baseline models, which are state-of-the-art.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval; I.2.6 [**Artificial Intelligence**]: *Learning*

General Terms

Algorithms, Experimentation

Keywords

Clickthrough Data, Latent Semantic Analysis, Topic Model, Linear Projection, Translation Model, Web Search

SIGIR'11, July 24-28, 2011, Beijing, P. R. China.

1. INTRODUCTION

Most modern search engines retrieve Web documents by literally matching terms in documents with those in a search query. However, lexical matching methods can be inaccurate due to the language discrepancy between Web documents and search queries [20, 31] i.e., a concept is often expressed using different vocabularies and language styles in documents and queries.

In the last two decades, different latent semantic models have been proposed to address the issue [e.g., 9, 19, 4]. Different terms that occur in a similar context are grouped into the same semantic cluster. Thus, a query and a document, represented as vectors in the lower-dimensional semantic space, can still have a high similarity even if they do not share any term.

An alternative strategy to cope with the problem is the approach based on statistical translation [2]: A query term can be a translation of any word in a document which may be different from, but semantically related to the query term; and the relevance of a document given a query is assumed proportional to the translation probability from the document to the query.

The research goal of this paper is to develop new ranking models for Web search by combining, in a principled way, the methods of semantic representation and statistical translation. Our assumption is that the translation between a query and a document can be modeled more effectively by mapping them into some semantic representations that are language independent than by mapping them at the word level.

Our work is based on two lines of previous research. The first is a set of clickthrough-based translation models for Web search presented and evaluated in [14], which is a significant extension of the original approach [2], motivated by the increasingly large amount of clickthrough data. Following [14], in this study we consider documents and queries as two different "languages" (i.e., the query language and the document language), and construct parallel training data from clickthrough data by pairing a query with the titles of the documents that have clicks for that query[1].

[1] In modern search engines, a Web document is described by multiple fields [14, 31], including title, body, anchor text etc. In our experiments, we only used the title field of a Web document for ranking. In addition to providing simplicity for fast experimentation, our decision is motivated by two factors, as described and empirically justified in [14]. First, titles are more similar to queries both in length and in vocabulary, making the translation model learning more effective. Second, the title field gives a good single-field retrieval result, although it is much shorter

Different from [14], our models are trained to capture the mapping relationships between terms in a document and terms in a query at the sematic level rather than at the word level.

The second line of previous work that lays the foundation of this study is the research on cross-lingual and multi-lingual latent semantic models [11, 25, 27]. In these earlier works, various extensions of Latent Semantic Analysis (LSA) or topic models are developed for applications such as cross-lingual IR [11] and retrieval of parallel Web pages [27]. In this study we investigate a variety of bilingual latent semantic models, which are the extensions and variants of these previous models. In particular, we focus the research on three aspects: (1) the methods of learning semantic representations from clickthrough data, (2) the approaches to incorporating the latent semantic models into proper IR frameworks, and (3) the evaluation of their effectiveness for Web search on a real world data set.

In this paper we present two new document ranking models for Web search, a bilingual topic model and a discriminative projection model. Both models are learned on clickthrough data. The topic model represents a document as a distribution of semantic topics, and is incorporated into the language modeling framework for IR by assuming that a query term is generated from a mixture of these topics. The bilingual topic model learns a semantic representation in such a way that each query and its paired titles, extracted from clickthrough data, share as much as possible the same topic fractions. Unlike the topic model, the projection model, similar to LSA, represents a document as a point in the semantic space. It fits naturally the IR framework based on vector space model (VSM). The relevance of a query and a document is computed as the (cosine) similarity between their vectors in the semantic space. Different from LSA and its variants, our model learns a projection matrix, which maps the term-vector of a document onto a lower-dimensional semantic space, using a supervised learning method. Inspired by the learning-to-rank framework [6], the projection matrix is learned *discriminatively* in such a way that the distance between a query and its paired title, both represented as vectors in a projected semantic space, is smaller than that between the query and the titles of other documents which have no clicks for that query. We evaluated the two new ranking models on the Web search task using a real world data set. Results show that they significantly outperform their corresponding baseline models, which are considered state-of-the-art.

In the rest of the paper, Section 2 reviews related work. Sections 3 and 4 describe in detail the bilingual topic model and the discriminative projection model, respectively. Section 5 presents the experiments. Section 6 concludes the paper.

2. PREVIOUS WORK

This section reviews previous approaches to bridging the lexical gap between queries and documents for IR.

2.1 Statistical Translation Models

The statistical translation based approach is an extension of the language modeling approach to IR [2, 28]. Each document is scored by the likelihood of it translating into a query. Let $\mathbf{q} = q_1 \dots q_{|\mathbf{q}|}$ be a query and $\mathbf{d} = d_1 \dots d_{|\mathbf{d}|}$ be a document. A word translation model [2] assumes that both $\mathbf{q}$ and $\mathbf{d}$ are bags of words, and that the translation probability of $\mathbf{q}$ given $\mathbf{d}$ is computed as

$$P(\mathbf{q}|\mathbf{d}) = \prod_{q \in \mathbf{q}} \sum_{w \in \mathbf{d}} P(q|w) P(w|\mathbf{d}) \quad (1)$$

where $P(w|\mathbf{d})$ is the unigram probability of word w in $\mathbf{d}$, and $P(q|w)$ is the probability of translating w into a query term q.

It is easy to verify that if we only allow a word to be translated into itself, Eq. (1) is reduced to the simple exact term matching that is used by traditional unigram language modeling approaches. To bridge the lexical gap between queries and documents, the translation based approach allows a document to translate any one of its words w to a different but semantically related query term with a nonzero probability.

Unlike latent semantic models, which will be reviewed shortly, the translation-based approach does not map different terms into latent semantic clusters but learns translation relationships directly between a term in a document and a term in a query. A major challenge of this approach is how to estimate the translation probabilities. The ideal training data would be a large amount of query-document pairs, in each of which the document is judged as relevant to the query. Due to the lack of such training data, [2] resorts to some *synthetic* query-document pairs, and [22] simply uses the title-document pairs as substitutes for training data. Since recently, with the growing availability of search logs, it is possible to mine implicit relevance judgments from clickthrough data, and to generate a large amount of *real* query-document pairs for translation model training [14]. Given enough training data, more sophisticated translation models such as phrase models and factored models have also been investigated [14, 23, 26].

2.2 Generative Topic Models

One of the first topic models widely used for IR is Probabilistic Latent Semantic Analysis (PLSA) [19]. Although Hofmann applied PLSA to IR in the VSM framework in the original paper, PLSA is in nature a generative model, and can be more straightforwardly incorporated into the language modeling framework, under which documents are ranked by their probabilities of generating a query. In PLSA, a query, viewed as a short document, is generated from a document using the following process. First, a multinomial distribution $\boldsymbol{\theta}$ of T topics for each document is selected as the most likely topic distribution for the document. Second, a latent topic z is picked for each query term with probability $P(z|\mathbf{d}, \boldsymbol{\theta}) = \theta_z$. Finally, a query term q is generated with probability $P(q|\boldsymbol{\phi}_Z)$, where $\boldsymbol{\phi}_z$ is the topic specific word distribution. Assuming that both a document and a query are bag-of-words, the probability of generating the query $\mathbf{q}$ from the document $\mathbf{d}$ is

$$P(\mathbf{q}|\mathbf{d}) = \prod_{q \in \mathbf{q}} \sum_{z} P(q|\boldsymbol{\phi}_z)\theta_z. \quad (2)$$

Notice that both Eq. (1) and (2) are in the form of a factored generative model where a query term is generated from a mixture of factors. While in translation models the factors are simply words in a document, in PLSA the factors are hidden topics.

Latent Dirichlet Allocation (LDA) [4] generalizes PLSA to a proper generative model and places Dirichlet priors over the parameters $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$. As a result, in LDA, instead of a single most likely topic vector $\boldsymbol{\theta}$ for a document, a posterior distribution over vectors $\boldsymbol{\theta}$ is used, where the prior is a conjugate Dirichlet prior

---

than other fields such as anchor and body; thus it can serve as a reasonable baseline in our experiments. Nevertheless, our methods are not limited to the title field, and can be easily applied to the multi-field description.

which is the same for all documents. So, in theory LDA overcomes some problems of PLSA such as overfitting and the issues regarding generating queries from unseen documents [4, 32]. However, whether the theoretical superiority of LDA can be translated into significant empirical improvement over PLSA on realistic applications, such as Web search, remains to be demonstrated. The effectiveness of LDA for IR is demonstrated in [32] without directly comparing it to PLSA. [17] clarifies the relationship between LDA and PLSA in the context of IR, and concludes that PLSA is a *maximum a posteriori* (MAP) estimated LDA model. [1] shows that MAP inference performs comparably to the best Bayesian inference methods for LDA. Therefore, in our experiments all the topic models are implemented as PLSA, or equivalently, LDA with MAP inference.

Recently, these topic models have been extended to handle cross-lingual or multi-lingual cases, where there are pairs or tuples of corresponding documents in different languages. For example, the Poly-Lingual Topic Model (PLTM) [25] is an extension to LDA that views documents in a tuple as having a shared topic distribution $\boldsymbol{\theta}$. Each of the documents in the tuple uses $\boldsymbol{\theta}$ to select topics z, but could use a different language-specific, word-topic-distribution $\boldsymbol{\phi}_z^L$ to generate words for the topics. Two additional models, Joint PLSA (JPLSA) and Coupled PLSA (CPLSA) are introduced in [27]. JPLSA is a variant of PLTM when documents of different languages share the same word-topic distribution, and MAP inference, instead of Gibbs sampling, is performed. CPLSA extends JPLSA by constraining corresponding documents to have similar fractions of words assigned to each topic according to the posterior distribution of topic assignments, instead of sharing the prior topic distributions. The bilingual topic models discussed in Section 3 are the variants and extensions to these previous models when we strive to effectively learn model parameters on click-through data for the application of Web search.

2.3 Linear Projection Models

One of the most well-known linear projection models for IR is LSA [9]. LSA models the whole document collection using a $n \times d$ document-term matrix $\mathbf{C}$, where n is the number of documents and d is the number of word types, and performs singular value decomposition (SVD) on $\mathbf{C}$. The k biggest singular values are then used to find the $d \times k$ projection matrix. Thus, a document represented by a d-dimensional term vector can be mapped to a k-dimensional concept vector.

Similar to topic models, LSA can also be extended to handle pairs or tuples of parallel or comparable documents. For example, Cross-Language LSA (CL-LSA) [11] applies LSA to concatenated comparable documents from different languages. Oriented Principal Component Analysis (OPCA) [12, 27] solves a generalized Eigen problem by introducing a noise covariance matrix to ensure that comparable documents can be projected closely. Canonical Correlation Analysis (CCA) [30] finds projections that maximize the cross-covariance between the projected vectors. In all these methods, the linear projection is learned without explicitly taking into account how documents are ranked. We will show that learning the projection matrix discriminatively on query-title pairs, using a cost function closely related to the measure of evaluating document ranking, leads to a much more effective model for Web search.

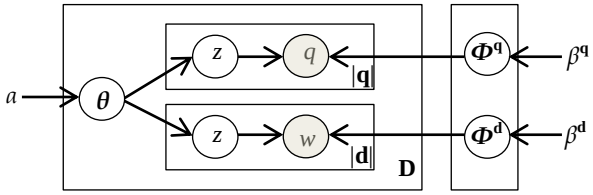

Figure 1: Graphical model for BLTM.

3. BILINGUAL TOPIC MODEL

The bilingual topic model (BLTM) can be viewed as a special case of PLTM [25], where search queries and web documents are assumed to be written in two different languages and MAP inference is used instead of Bayesian inference. BLTM is also a close variant of JPLSA [27].

We assume that a query $\mathbf{q} = q_1 \dots q_{|\mathbf{q}|}$ and its paired title $\mathbf{d} = w_1 \dots w_{|\mathbf{d}|}$ share a common distribution of topics, but use different (probably overlapping) vocabularies to express these topics. The graphical model is shown in Figure 1. Formally, BLTM assumes the following process of generating a query-title pair.

- First, for each topic z, a pair of different word distributions $(\boldsymbol{\phi}_z^{\mathbf{q}}, \boldsymbol{\phi}_z^{\mathbf{d}})$ are selected from a Dirichlet prior with concentration parameter β, where $\boldsymbol{\phi}_z^{\mathbf{q}}$ is a topic-specific query-word distribution, and $\boldsymbol{\phi}_z^{\mathbf{d}}$ a topic-specific title-word distribution. Assuming there are T topics, we have two sets of distributions $\boldsymbol{\Phi}^{\mathbf{q}} = (\boldsymbol{\phi}_1^{\mathbf{q}}, \dots, \boldsymbol{\phi}_T^{\mathbf{q}})$ and $\boldsymbol{\Phi}^{\mathbf{d}} = (\boldsymbol{\phi}_1^{\mathbf{d}}, \dots, \boldsymbol{\phi}_T^{\mathbf{d}})$.
- Then, for each query and its paired title, a topic distribution $\boldsymbol{\theta}^{\mathbf{q},\mathbf{d}}$ is drawn from a Dirichlet prior with concentration parameter α.
- Each term in the query is then generated by first selecting a topic z according to $\boldsymbol{\theta}^{\mathbf{q},\mathbf{d}}$, and drawing a word from $\boldsymbol{\phi}_z^{\mathbf{q}}$.
- Similarly, each term in the paired title is generated by selecting a topic z according to the same topic distribution $\boldsymbol{\theta}^{\mathbf{q},\mathbf{d}}$, and then drawing a word from $\boldsymbol{\phi}_z^{\mathbf{d}}$.

Thus, the log-likelihood of a corpus of query-title pairs, together with the paired document-topic vectors and word-topic vectors, is

$$\log\left(P(\boldsymbol{\Phi}^{\mathbf{q}}|\beta^{\mathbf{q}}) P(\boldsymbol{\Phi}^{\mathbf{d}}|\beta^{\mathbf{d}}) \prod_{(\mathbf{q},\mathbf{d})} P(\boldsymbol{\theta}|\alpha) P\big((\mathbf{q},\mathbf{d})\big|\boldsymbol{\theta}^{\mathbf{q},\mathbf{d}}, \boldsymbol{\Phi}^{\mathbf{q}}, \boldsymbol{\Phi}^{\mathbf{d}}\big) \right) \quad (3)$$

where

$$P\big((\mathbf{q},\mathbf{d})\big|\boldsymbol{\theta}^{\mathbf{q},\mathbf{d}}, \boldsymbol{\Phi}^{\mathbf{q}}, \boldsymbol{\Phi}^{\mathbf{d}}\big) = \prod_{q\in\mathbf{q}} \sum_z P\big(q\big|\,\boldsymbol{\phi}_z^{\mathbf{q}}\big) P\big(z\big|\boldsymbol{\theta}^{\mathbf{q},\mathbf{d}}\big) \cdot \prod_{w\in\mathbf{d}} \sum_z P\big(w\big|\,\boldsymbol{\phi}_z^{\mathbf{d}}\big) P\big(z\big|\boldsymbol{\theta}^{\mathbf{q},\mathbf{d}}\big)$$

Note that since we use MAP estimation, as will be described in Section 3.1, $\boldsymbol{\theta}^{\mathbf{q},\mathbf{d}}$, $\boldsymbol{\Phi}^{\mathbf{q}}$ and $\boldsymbol{\Phi}^{\mathbf{d}}$ in Eq. (3) are treated as parameters rather than hidden variables, as in the Bayesian inference methods.

3.1 MAP Estimation

We use the standard EM algorithm [10] to estimate the parameters $(\boldsymbol{\theta}^{\mathbf{q},\mathbf{d}}, \boldsymbol{\Phi}^{\mathbf{q}}, \boldsymbol{\Phi}^{\mathbf{d}})$ of BLTM by maximizing the joint log-likelihood of the parallel corpus and the parameters, as shown in Eq. (3). The derivation of the updates is similar to that described in [7, 8]. In the E-step, the posterior probabilities for each term q in query $\mathbf{q}$

and each term w in its paired title $\mathbf{d}$ are computed for the latent variables z according to:

$$P(z|q,\boldsymbol{\theta}^{\mathbf{q},\mathbf{d}}) = \frac{P(q|\boldsymbol{\phi}_z^{\mathbf{q}})P(z|\boldsymbol{\theta}^{\mathbf{q},\mathbf{d}})}{\sum_{z'} P(q|\boldsymbol{\phi}_{z'}^{\mathbf{q}})P(z'|\boldsymbol{\theta}^{\mathbf{q},\mathbf{d}})} \quad (4)$$

$$P(z|w,\boldsymbol{\theta}^{\mathbf{q},\mathbf{d}}) = \frac{P(w|\boldsymbol{\phi}_z^{\mathbf{d}})P(z|\boldsymbol{\theta}^{\mathbf{q},\mathbf{d}})}{\sum_{z'} P(w|\boldsymbol{\phi}_{z'}^{\mathbf{d}})P(z'|\boldsymbol{\theta}^{\mathbf{q},\mathbf{d}})} \quad (5)$$

In the M-step, parameters are updated for given posterior probabilities computed in the previous E-step. We treat α, $\beta^{\mathbf{d}}$ and $\beta^{\mathbf{q}}$ as hyperparameters, each corresponding to one Dirichlet prior, and denote Q as the size of the query vocabulary and W the size of title vocabulary. To simplify notation, letting $n(q,\mathbf{q})$ be the frequency of term q in query $\mathbf{q}$, and $n(w,\mathbf{d})$ the frequency of term w in title $\mathbf{d}$, we define ${N_{q,z}}^{\mathbf{q},\mathbf{d}} = n(q,\mathbf{q})P(z|q,\boldsymbol{\theta}^{\mathbf{q},\mathbf{d}})$ and ${N_{w,z}}^{\mathbf{q},\mathbf{d}} = n(w,\mathbf{d})P(z|w,\boldsymbol{\theta}^{\mathbf{q},\mathbf{d}})$. Then, the updates can be written as

$$P(q|\boldsymbol{\phi}_z^{\mathbf{q}}) = \frac{\beta^{\mathbf{q}} - 1 + \sum_{(\mathbf{q},\mathbf{d})} {N_{q,z}}^{\mathbf{q},\mathbf{d}}}{Q\beta^{\mathbf{q}} - Q + \sum_{(\mathbf{q},\mathbf{d}),q'} {N_{q',z}}^{\mathbf{q},\mathbf{d}}} \quad (6)$$

$$P(w|\boldsymbol{\phi}_z^{\mathbf{d}}) = \frac{\beta^{\mathbf{d}} - 1 + \sum_{(\mathbf{q},\mathbf{d})} {N_{w,z}}^{\mathbf{q},\mathbf{d}}}{W\beta^{\mathbf{d}} - W + \sum_{(\mathbf{q},\mathbf{d}),w'} {N_{w',z}}^{\mathbf{q},\mathbf{d}}} \quad (7)$$

$$P(z|\boldsymbol{\theta}^{\mathbf{q},\mathbf{d}}) = \frac{\alpha - 1 + (\sum_q {N_{q,z}}^{\mathbf{q},\mathbf{d}} + \sum_w {N_{w,z}}^{\mathbf{q},\mathbf{d}})}{T\alpha - T + \sum_{z'}(\sum_q {N_{q,z'}}^{\mathbf{q},\mathbf{d}} + \sum_w {N_{w,z'}}^{\mathbf{q},\mathbf{d}})} \quad (8)$$

3.2 Posterior Regularization (PR)

A query and a title, if they are paired, are expected to not only share the same prior distribution over topics but also contain similar fractions of words assigned each topic. Since MAP estimation of the shared topic vector is concerned with explaining the union of tokens in the query and document and can be easily dominated by the longer one of the two, it does not guarantee that each topic z occurs with similar frequency in the query and title. Thus, following [27], we extend BTLM by constraining the paired query and title to have similar fractions of tokens assigned to each topic, and the constraint is enforced on expectation using posterior regularization [13].

BTLM with posterior regularization (BTLM-PR) is a variant of CPLSA [27] with two important modifications. First, while BTLM-PR assumes a pair of query and title to share the same topic distribution $\boldsymbol{\theta}$ as shown in Figure 1, in CPLSA the topic distributions of a pair of documents in two languages are completely independent. Second, CPLSA uses inequality constraints with slack variables to form the constrained space whereas BTLM-PR uses only equality constraints. These modifications not only lead to a mathematically simpler model, thus making the model training faster (e.g., it requires fewer EM iterations), but also significantly improve the retrieval results.

BTLM-PR can be trained using a modified EM algorithm, where in the E-step the posterior distributions of topics computed on a query-title pair $(\mathbf{q},\mathbf{d})$ are *projected* onto a constrained set of distributions, for which the expected fraction of tokens in $\mathbf{q}$ that are assigned topic t is the same as the expected fraction of tokens in $\mathbf{d}$ that are assigned the same topic. The expected counts with respect to this projected posterior distribution are then used in the M-step, which remains the same as Eq. (6) to (8).

In what follows, we describe in detail how the projection is performed. Let $(\mathbf{q},\mathbf{d})$ be a pair of sequences of tokens and their topic assignments, where

$$\mathbf{q} = \{(q_1,\dots,q_{|\mathbf{q}|}),(z_1,\dots,z_{|\mathbf{q}|})\}$$
$$\mathbf{d} = \{(w_1,\dots,w_{|\mathbf{d}|}),(z_1,\dots,z_{|\mathbf{d}|})\}.$$

The projection minimizes the Kullback-Leibler divergence between two sets of distributions $\boldsymbol{P}$ and $\boldsymbol{P}'$, $\mathbf{KL}(\boldsymbol{P}'||\boldsymbol{P})$. Let $\boldsymbol{P}$ denote the set of posterior distributions over hidden topic assignments for the tokens in $\mathbf{q}$ and $\mathbf{d}$, computed by the standard E-step as in Eq. (4) and (5). There are $|\mathbf{q}|$ plus $|\mathbf{d}|$ distributions in this set.

$$\boldsymbol{P} = \{P(z|q,\boldsymbol{\theta}^{\mathbf{q},\mathbf{d}}),P(z|w,\boldsymbol{\theta}^{\mathbf{q},\mathbf{d}})\}.$$

$\boldsymbol{P}'$ is an ideal set of distributions that has the desired property: the expected fraction of each topic is equal in $\mathbf{q}$ and $\mathbf{d}$

$$\boldsymbol{P}' = \{{P_{\mathbf{q}}}'(z|q),{P_{\mathbf{d}}}'(z|w)\},$$

such that for each topic

$$\mathbf{E}_{\boldsymbol{P}'_{\mathbf{q}}}\left[\frac{1}{|\mathbf{q}|}\sum_{j=1}^{|\mathbf{q}|}\mathbf{1}(z_j = t)\right] = \mathbf{E}_{\boldsymbol{P}'_{\mathbf{d}}}\left[\frac{1}{|\mathbf{d}|}\sum_{j=1}^{|\mathbf{d}|}\mathbf{1}(z_j = t)\right],$$

where

$$\mathbf{E}_{\boldsymbol{P}'_{q}}\left[\frac{1}{|\mathbf{q}|}\sum_{j=1}^{|\mathbf{q}|}\mathbf{1}(z_j = t)\right] = \frac{1}{|\mathbf{q}|}\sum_{q\in\mathbf{q}} P'(z=t|q)\text{, and}$$

$$\mathbf{E}_{\boldsymbol{P}'_{d}}\left[\frac{1}{|\mathbf{d}|}\sum_{j=1}^{|\mathbf{d}|}\mathbf{1}(z_j = t)\right] = \frac{1}{|\mathbf{d}|}\sum_{w\in\mathbf{d}} P'(z=t|w).$$

Now, the projection can be formulated as a constrained optimization problem, where we seek an ideal set of distributions $\boldsymbol{P}'$ that is closest to $\boldsymbol{P}$:

$$\min_{\boldsymbol{P}'\in\boldsymbol{\mathcal{P}}} \mathbf{KL}(\boldsymbol{P}'||\boldsymbol{P}) \quad (9)$$

where $\boldsymbol{\mathcal{P}}$ is a space of distribution sets $\boldsymbol{P}'$ satisfying the constraint:

$$\mathbf{E}_{\boldsymbol{P}'_{q}}\left[\frac{1}{|\mathbf{q}|}\sum_{j=1}^{|\mathbf{q}|}\mathbf{1}(z_j = t)\right] = \mathbf{E}_{\boldsymbol{P}'_{d}}\left[\frac{1}{|\mathbf{d}|}\sum_{j=1}^{|\mathbf{d}|}\mathbf{1}(z_j = t)\right], t = 1\dots T.$$

In our case, the valid ideal distribution space $\boldsymbol{\mathcal{P}}$ is non-empty. The problem of Eq. (9) can be solved efficiently in its dual form. For the sake of clarity, we leave the derivation to Appendix A, and only present the result below. The primal solution P' is given in terms of the dual solution λ_t by

$$P'(z=t|q) = \frac{1}{Z_{\lambda,q}}P(z=t|q,\boldsymbol{\theta}^{\mathbf{q},\mathbf{d}})\exp\left(-\frac{\lambda_t}{|\mathbf{q}|}\right) \quad (10)$$

where $Z_{\lambda,q} = \sum_t P(z=t|q,\boldsymbol{\theta}^{\mathbf{q},\mathbf{d}})\exp\left(-\frac{\lambda_t}{|\mathbf{q}|}\right)$, and

$$P'(z=t|w) = \frac{1}{Z_{\lambda,w}}P(z=t|w,\boldsymbol{\theta}^{\mathbf{q},\mathbf{d}})\exp\left(\frac{\lambda_t}{|\mathbf{d}|}\right) \quad (11)$$

where $Z_{\lambda,w} = \sum_t P(z=t|w,\boldsymbol{\theta}^{\mathbf{q},\mathbf{d}})\exp\left(\frac{\lambda_t}{|\mathbf{d}|}\right)$.

The optimal value of λ_t can be obtained using the gradient ascent algorithm with the update

$$\lambda_t^{new} = \lambda_t^{old} + \epsilon\cdot\mathbf{g}(\lambda_t) \quad (12)$$

where ϵ is the learning rate, and the gradient $\mathbf{g}$ is computed as

$$\mathbf{g}(\lambda_t) = \mathbf{E}_{P'_{\mathbf{q}}}\left[\frac{1}{|\mathbf{q}|}\sum_{j=1}^{|\mathbf{q}|}\mathbf{1}(z_j = t)\right] - \mathbf{E}_{P'_{\mathbf{d}}}\left[\frac{1}{|\mathbf{d}|}\sum_{j=1}^{|\mathbf{d}|}\mathbf{1}(z_j = t)\right] \quad (13)$$

In summary, the projection step is performed for each pair $(\mathbf{q}, \mathbf{d})$ in training data as follows. First, parameters $\lambda_t, t = 1 \ldots T$ are optimized using gradient ascent according to Eq. (12) and (13). Then, the projected probabilities are computed according to Eq. (10) and (11). The modified EM algorithm uses the projected posterior probabilities P' to update the model parameters in the M-step.

3.3 Ranking Documents

In our Web search experiments, we mixed BLTM with the smoothed unigram language model, and used the following document ranking function:

$$P_s(\mathbf{q}|\mathbf{d}) = \prod_{q\in\mathbf{q}} P_s(q|\mathbf{d}) \quad (14)$$

$$P_s(q|\mathbf{d}) = \lambda_1 P(q|C) + (1-\lambda_1)P_{mx}(q|\mathbf{d}) \quad (15)$$

$$P_{mx}(q|\mathbf{d}) = \lambda_2 P(q|\mathbf{d}) + (1-\lambda_2)P_{bltm}(q|\mathbf{d}) \quad (16)$$

$$P_{bltm}(q|\mathbf{d}) = \sum_z P\left(q|\boldsymbol{\phi}_z^{\boldsymbol{q}}\right)P(z|\boldsymbol{\theta}^{\boldsymbol{d}}) \quad (17)$$

Here, $P(q|C)$ in Eq. (15) and $P(q|\mathbf{d})$ in Eq. (16) are the unsmoothed background model and document model, respectively. λ_1 and λ_2 are tuning parameters with their values between 0 and 1. Notice that letting $\lambda_2 = 1$ reduces the model to a unigram language model with Jelinek-Mercer smoothing [34], which is used as baseline in our experiments. Letting $\lambda_2 = 0$, the document model depends solely on BLTM. Also notice that BLTM in Eq. (17) differs from a topic model of Eq. (2). Although in both models the topics are generated from $\mathbf{d}$ written in *title language*, the query term q is generated from topic-specific word distributions in *query language* in BLTM. Thus, BLTM can be considered as performing a translation from title to query via hidden topics. In our experiments, we used folding-in with 20 EM iterations to map a document in test data to its corresponding topic vector $\boldsymbol{\theta}^{\boldsymbol{d}}$.

4. DISCRIMINATIVE PROJECTION MODEL

The discriminative projection model (DPM) maps a sparse, high-dimensional term vector onto a dense, low-dimensional space through a simple matrix multiplication. DPM differs from other linear projection models in the way the projection matrix is learned. In this study we compare DPM to LSA and its variants. Since in our implementation DPM and various LSA models take the same clickthrough data as input and output the projection matrix in the same model form, we start the description of DPM with a brief review of LSA and its variants.

4.1 LSA and its Variants

LSA models the whole document collection using a $n \times d$ document-term matrix $\mathbf{C}$, where n is the number of documents and d is the number of word types. $\mathbf{C}$ is first factored into the product of three matrices using SVD

$$\mathbf{C} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T \quad (18)$$

where the orthogonal matrices $\mathbf{U}$ and $\mathbf{V}$ are called term and document vectors, respectively, and the diagonal elements of $\boldsymbol{\Sigma}$ are singular values in descending order. Then, a low-rank matrix approximation of $\mathbf{C}$ is generated by retaining only the k biggest singular values in $\boldsymbol{\Sigma}$. Now, a document (or a query) represented by a term vector $\mathbf{D}$ can be mapped to a low-dimensional concept vector $\widehat{\mathbf{D}}$ as

$$\widehat{\mathbf{D}} = \mathbf{A}^T\mathbf{D} \quad (19)$$

where the $d \times k$ matrix $\mathbf{A} = \mathbf{U}_k\boldsymbol{\Sigma}_k^{-1}$ is called the projection matrix. In document search, the relevance score between a query and a document, represented respectively by term vectors $\mathbf{Q}$ and $\mathbf{D}$, is assumed to be proportional to their cosine similarity score of the corresponding concept vectors $\widehat{\mathbf{Q}}$ and $\widehat{\mathbf{D}}$, according to the projection matrix $\mathbf{A}$

$$sim_{\mathbf{A}}(\mathbf{Q}, \mathbf{D}) = \frac{\widehat{\mathbf{Q}}\widehat{\mathbf{D}}}{\|\widehat{\mathbf{Q}}\|\|\widehat{\mathbf{D}}\|} \quad (20)$$

Notice that LSA is closely related to principal component analysis (PCA), and can be solved via Eigen-decomposition instead. Let $\mathbf{S}$ be the correlation matrix between terms $\mathbf{S} = \mathbf{C^T C}$. The projection matrix $\mathbf{A}$ is exactly the top-k Eigen vectors of $\mathbf{S}$. Compared to LSA, the only difference is that PCA solves the Eigen problem on the covariance matrix instead. Because the term vectors are very sparse in practice and the column means are close to zero, $\mathbf{S}$ is in fact very close to the covariance matrix. In our experiments we used PCA to derive the LSA projection matrix.

The objective of LSA is to find the matrix to maximize the variance of the projected vectors. When applied directly to the clickthrough data, queries and title are treated as separate documents and the click information is not used. CL-LSA [11] and OPCA [12, 27] are two variants to LSA, striving to reduce the projected distance between a query and its paired title. Unfortunately, although both variants use the pair information to bias the derivation of the projection matrix, their objective functions are in fact only a coarse approximation to the final ranking measure. We hypothesize that with a large set of training data, the projection matrix can be learned discriminatively by minimizing a loss function that targets the ranking scenario, and thus yields better results. We describe this new projection learning framework in the next section.

4.2 Projection Learning via Siamese Neural Network (S2Net)

The projection matrix in DPM is learned from query-title pairs using S2Net, a newly proposed learning framework that learns discriminatively the projection matrix from pairs of related and unrelated documents. We briefly introduce the model below and interested readers can refer to [33] for more detail.

S2Net treats the raw term vector as the input layer and the mapped concept vector as the output layer. The value of each node in the output layer is a linear sum of all the input nodes, where the weights are associated with the edges. In other words, the network structure is a complete bipartite graph between the input and output layers, and the edge weights are equivalent to the form of a linear projection matrix $\mathbf{A}$, as in Eq. (19). Below, we describe the loss function and training process.

The design of the loss function in S2Net follows the pairwise learning-to-rank paradigm outlined in [6]. Consider a query $\mathbf{q}$ and two documents $\mathbf{d_1}$ and $\mathbf{d_2}$, where $\mathbf{d_1}$ has clicks for $\mathbf{q}$ but $\mathbf{d_2}$ does not. Let $\mathbf{Q}, \mathbf{D}_1$ and $\mathbf{D}_2$ be the term vectors of $\mathbf{q}, \mathbf{d_1}$ and $\mathbf{d_2}$, respec-

tively. We then construct two pairs of term vectors $(\mathbf{Q}, \mathbf{D}_1)$ and $(\mathbf{Q}, \mathbf{D}_2)$, where the former is preferred and should be ranked higher. Given the model (i.e., the projection matrix) $\mathbf{A}$, let Δ be the difference of the cosine similarity scores of their projected concept vectors, following Eq. (20). Namely, $\Delta = sim_{\mathbf{A}}(\mathbf{Q}, \mathbf{D}_1) - sim_{\mathbf{A}}(\mathbf{Q}, \mathbf{D}_2)$. Intuitively, we want to learn a model to increase Δ. We use the following logistic loss over Δ, which can be shown to upper bound the pairwise accuracy

$$L(\Delta; \mathbf{A}) = \log(1 + \exp(-\gamma\Delta)) \quad (21)$$

The loss function in Eq. (21) has a shape similar to the hinge loss used in SVMs. Because of the use of the cosine similarity function, we add a scaling factor γ that magnifies Δ from [-2, 2] to a larger range. Empirically, the value of γ makes no difference as long as it is large enough. In the experiments, we set $\gamma = 10$. Because the loss function is differentiable, optimizing the model parameters $\mathbf{A}$ can be done using the gradient-based methods, such as L-BFGS. For the sake of a clean presentation, we leave the gradient derivation to Appendix B.

Given that the optimization problem is not convex, initializing the model from a good projection matrix often helps reduce the training time and may converge to a better local minimum. In our experiments, we always started the model parameters from the LSA matrix. In principle, Eq. (21) can further be regularized by adding a term $\frac{\beta}{2}\|\mathbf{A} - \mathbf{A}_0\|^2$, which forces the learned model not to deviate too much from the initial model $\mathbf{A}_0$. However, we did not find clear empirical advantage over the simpler *early stop* approach in a preliminary study, which is adopted in the experiments in this paper.

5. EXPERIMENTS

This section evaluates the effectiveness of the models described in Sections 3 and 4 on the Web search application. Instead of presenting a direct comparison between topic modeling and linear projection modeling[2], we focus our experiments on demonstrating that for each type of models, clickthrough data can lead to significant improvements when modeled properly. Thus, we will report the results of the topic models and the linear projection models in separate sections. For each type, we compare our models to their baseline methods, which are considered state-of-the-art in the research community.

5.1 Data Sets and Evaluation Methodology

We evaluated the retrieval models on a large-scale real world data set, called the evaluation data set henceforth. The data set contains 16,510 English queries sampled from one-year query log files of the Microsoft Bing search engine. On average, each query is associated with 15 Web documents (URLs). Each query-title pair has a relevance label. The label is human generated and is on a 5-level relevance scale, 0 to 4, with 4 meaning document $\mathbf{d}$ is the most relevant to query $\mathbf{q}$ and 0 meaning $\mathbf{d}$ is not relevant to $\mathbf{q}$. All the queries and documents are preprocessed as follows. The text is white-space tokenized and lowercased, numbers are retained, and no stemming/inflection is performed.

All the ranking models used in this study (i.e., language models, topic models, VSM and linear projection models) contain free parameters that must be estimated empirically by trial and error. Therefore, we used 2-fold cross validation: A set of results on one half of the data is obtained using the parameter settings optimized on the other half, and the global retrieval results are combined from those of the two sets.

The performance of all the ranking models was measured by mean *Normalized Discounted Cumulative Gain* (NDCG) [21]. We report NDCG scores at truncation levels 1, 3, and 10. We also performed a significance test using the paired *t*-test. Differences are considered statistically significant when the *p*-value is less than 0.05.

In our experiments, the query-title pairs, used for model training, are extracted from one year query log files using a procedure similar to [16]. First of all, a set of query sessions were extracted from the raw log files. A query session consists of a user-issued query and a ranked list of documents, each of which may or may not be clicked by a user. Second, we built for each document a so-called query click field, which consists of a set of query-score pairs $(\mathbf{q}, Score(\mathbf{d}, \mathbf{q}))$, where $\mathbf{q}$ is a unique query string for which the document $\mathbf{d}$ has clicks and $Score(\mathbf{d}, \mathbf{q})$ is a score assigned to $\mathbf{q}$, as defined in Eq. (1) in [16]. Only those pairs whose scores are larger than a threshold were retained. $Score(\mathbf{d}, \mathbf{q})$ could be the number of times the document was clicked on for that query, but it is important to also consider the number of times the page has been shown to the user and the position in the ranked list at which the page was shown. Finally, we formed a set of query-title pairs by aligning the title of the document to each unique query string in the query click field of the same document.

Some previous studies [e.g., 16, 29] show that the query click field, when it is valid, is the most effective for Web search. However, click information is unavailable for many URLs, especially new URLs and tail URLs, leaving their click fields invalid (i.e., the field is either empty or unreliable because of sparseness). In this study, we assume that each document contained in the evaluation data set is either a new URL or a tail URL, thus has no click information (i.e., its click field is invalid). Our research goal is to investigate how to learn the latent semantic models from the popular URLs that have rich click information, and apply the models to improve the retrieval of those tail or new URLs. To this end, in our experiments only the title fields of the Web documents are used for ranking.

From one-year query log files, we were able to generate large amounts of query-title pairs using the procedure described above. For training latent semantic models in this study, we used a randomly sampled subset of 82,834,648 pairs whose documents are popular and have rich click information. We then tested the trained models in ranking the documents in the evaluation data set, which do not have click information. The query-title pairs were pre-processed in the same way as the evaluation data to ensure uniformity.

[2] Strictly speaking, the results of topic models and linear projection models, reported in Tables 1 and 2 respectively, cannot be compared directly due to three reasons. First, the baseline models, with which the proposed models are combined to achieve the best results, are different. Second, in order to perform S2Net training efficiently for fast experimentation, the vocabulary used in building linear projection models is much smaller than that for topic models. Third, the input term weighting functions for them are different: the topic models use the raw term frequency counts, while the projection models take TFIDF vectors. We leave to future work a direct comparison of topic modeling and linear projection modeling in a more consistent setting, and how to best combine them for Web document ranking.

5.2 Topic Model Results

Table 1 shows the main Web document ranking results using various topic models, tested on the human-labeled evaluation data set via 2-fold cross validation, as described in Section 5.1.

UM (Row 1) is the baseline model, a unigram language model with Jelinek-Mercer smoothing, parameterized by Eq. (14) to (16) with $\lambda_2 = 1$.

Rows 2 to 9 are four different topic models, parameterized by Eq. (14) to (17). To improve the efficiency of model training, we pruned the query-title training data by retaining only top 500K high-frequency words. We used 100 topics ($T = 100$) for all the topic models. In order to investigate the relative contributions of the unigram model and the latent semantic model to ranking, we report for each topic model the results using two different settings. One is letting the document model solely depend on the latent semantic model by setting $\lambda_2 = 0$ in Eq. (16). These results are shown in the shaded rows in Table 1. The other is defining the document model as a mixture of the unigram model and the latent semantic model by using a nonzero λ_2 in Eq. (16), tuned via cross-validation. We used folding-in with 20 EM iterations to map each document in the evaluation data set to its corresponding topic vector. In what follows, we describe the four topic models in turn.

PLSA (Rows 2 and 3) is our implementation of the model proposed in [19], and was trained on documents only (i.e., the title side of the query-title pairs). Different from [19], our version of PLSA was learned using MAP estimation, with $\alpha = 1.1$ and $\beta = 1.01$. The model can also be viewed as an approximation to the LDA document model described in [32], which is learned on the TREC document collection via Gibbs sampling.

BLTM (Rows 4 and 5) is the model described in Figure 1, where the model parameters were learned on query-title pairs using MAP estimation, as described in 3.1. We found that the model performance is not very sensitive to the values of the hyperparameters, which were set in our experiments as $\alpha = 1.1$ and $\beta^d = \beta^q = 1.01$. We also found after around 20 EM iterations, the likelihood of the model barely increases.

BLTM-PR (Rows 6 and 7) is BLTM trained using the modified EM algorithm that uses posterior regularization (PR), described in Section 3.2, to constrain the paired query and title not only to share the same prior topic distribution $\boldsymbol{\theta}$, but to also have similar fractions of tokens assigned to each topic. We found that with PR, the EM algorithm takes fewer iterations to converge. In our experiments, the likelihood seems to saturate after 16 iterations.

BLTM-PR-1V (Rows 8 and 9) is a variant of BLTM-PR where we merge the vocabularies in query and title languages and learn topic-specific word distributions over these merged vocabularies. This is suggested in [27], arguing that such a variant not only simplifies the implementation but also sometimes leads to better results. In our experiments, we found that using the merged vocabularies does not bring any significant difference for BLTM, but does lead to some small but significant improvement for BLTM-PR. One possible interpretation is that the same word in a query and document very often has the same topic, which is not used at all by the two-vocabulary version. In particular, for some rare words that may be harmful since there might not be enough data to estimate their topic distributions in queries and titles completely independently. Therefore, we speculate that a hierarchical Bayesian model that encourages matching words in queries and titles to have the same topics, but also allows them to diverge, would be superior to both the single vocabulary and two-vocabulary models. We leave it to future work.

#	Models	NDCG@1	NDCG@3	NDCG@10
1	**UM**	0.308	0.373	0.454
2	**PLSA** ($\lambda_2 = 0$)	0.295	0.371	0.456
3	**PLSA**	0.325	0.391	0.470
4	**BLTM** ($\lambda_2 = 0$)	0.330	0.399	0.476
5	**BLTM**	0.338	0.404	0.479
6	**BLTM-PR** ($\lambda_2 = 0$)	0.334	0.403	0.479
7	**BLTM-PR**	0.342	0.406	0.482
8	**BLTM-PR-1V** ($\lambda_2 = 0$)	0.337	0.403	0.480
9	**BLTM-PR-1V**	0.344	0.407	0.483
10	**WTM_M1** ($\lambda_2 = 0$)	0.332	0.400	0.478
11	**WTM_M1**	0.338	0.404	0.480

Table 1: Web document ranking results using different topic models, tested on the evaluation data set, where only the title field of each document is used.

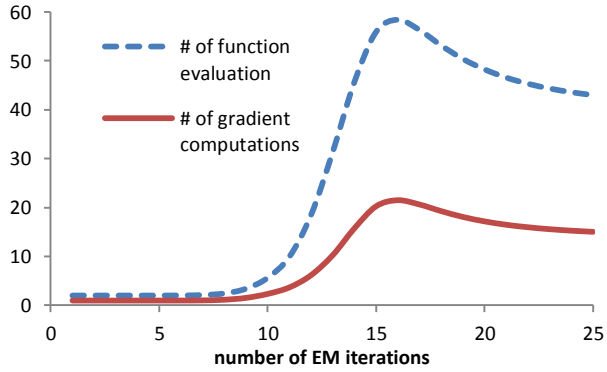

Figure 2: Average number of function evaluations and gradient computations per EM iteration, as a function of the number of the EM iterations, in the projection step for training BLTM-PR.

WTM_M1 (Rows 10 and 11) is our implementation of the word translation model described in [14], listed here for comparison. The corresponding ranking function is similar to Eq. (14) to (16). The only difference is that P_{bltm} in Eq. (16) is replaced by the word translation model P_{wtm} defined as

$$P_{wtm}(q|\mathbf{d}) = \sum_{w \in \mathbf{d}} P(q|w)P(w|\mathbf{d}),$$

where $P(q|w)$ is the word translation probability assigned by IBM-Model-1 [5], trained on query-title pairs using EM.

The results in Table 1 suggest several conclusions. First, using PLSA alone as a document model hurts the ranking performance (Row 2 vs. Row 1). But a linear combination of PLSA and the original document model significantly outperforms the baseline model (Row 3 vs. Row 1). The results are consistent with those previously reported on the TREC collections [32]. Second, using clickthrough data for model training by extending PLSA to BLTM, leads to a significant improvement (Rows 4 and 5 vs. Rows 2 and 3). Third, the performance of BLTM can be further improved by introducing constraints in the EM training to force the paired query and title to share the same proportion of topics (Rows 6 to 9 vs. Rows 4 and 5). The differences among BLTM, BLTM-PR, and BLTM-PR-V1 are statistically significant. Finally, we confirmed the effectiveness of the word translation model. The

model performs as well as BLTM, i.e., their results are not significantly different (Rows 10 and 11 vs. Rows 4 and 5). However, both BLTM-PR and BLTM-PR-V1 beat the translation model with a statistically significant margin (Rows 6 to 9 vs. Rows 10 and 11). We also tried combining WTM_M1 with BLTM-PR, but the result is not significantly better than that of BLTM-PR.

The complexity of the training algorithm for BLTM is the same as the EM training for PLSA, which has been well-studied. BLTM-PR uses a modified EM algorithm. Although BLTM-PR needs fewer EM iterations to converge, each iteration is more expensive due to the projection step. The runtime of the projection step is dominated by function evaluations (Eq. (28)), and otherwise the most expensive step is the computation of the gradients (Eq. (13)). Notice that the projection function needs to be called for each query-title pair. Initializing BLTM-PR with a uniform distribution for $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$, Figure 2 plots the average number of function evaluations and gradient computations per EM iteration, as a function of the number of the EM iterations. Both curves show that after 10 EM iterations, the training becomes much slower due to the dramatically increased cost of the projection step, indicating that from this moment EM starts to lead the distribution set far away from the ideal one in terms of KL distance. The cost of projection reduces slightly after 16 iterations when the EM training saturates. In our experiments, we found that training BLTM takes around 30 hours on a commodity 8-core server with 64-GB memory, and training BLTM-PR takes twice as much time. In practice, since the EM algorithm can be easily parallelized, topic model training could be performed much more efficiently on a cluster of computers.

5.3 Linear Projection Model Results

Table 2 shows the main Web document ranking results using various linear projection models, tested on the human-labeled evaluation data set.

VSM (Row 1) is the baseline model, where both documents and queries are represented as term vectors, with the TF-IDF term weighting, and the documents are ranked by the cosine similarity between the query and document vectors.

Rows 2 to 9 are four different linear projection models. All of them have the same model form as that of LSA [9]. To improve the efficiency of model training, we truncated the term vectors based on a vocabulary consisting of only the top 40K high document-frequency (DF) words, where the DF values are calculated based on the clickthrough data. We used 100 dimensions (k=100) for the vectors in semantic space. Similar to topic models, we report a pair of results for each projection model using two different settings. One is ranking documents using the cosine similarity scores in the semantic space, as in Eq. (20). These results are shown in the shaded rows in Table 2. The other is ranking documents based a weighted linear combination of two cosine similarity scores, computed in the original term space and in the projected semantic space, respectively. The linear combination weight is tuned via cross-validation. In what follows, we describe each of these models in turn.

LSA (Rows 2 and 3) is our implementation of the model described in [9]. As described in Section 4.1, we used PCA instead of SVD to compute the matrix. Queries and titles are treated as separate documents; the pair information from the clickthrough data was not used in this model.

CL-LSA (Rows 4 and 5) extends LSA by leveraging the pair information so that the projected distance between a query and its paired title is reduced [11]. In our implementation, each query and its paired title were concatenated first to form a new document. Then, the projection matrix was learned by applying LSA to this new corpus.

#	Models	NDCG@1	NDCG@3	NDCG@10
1	**VSM**	0.313	0.379	0.460
2	**LSA**	0.298	0.372	0.455
3	**LSA + VSM**	0.330	0.396	0.474
4	**CL-LSA**	0.298	0.370	0.454
5	**CL-LSA + VSM**	0.330	0.396	0.474
6	**OPCA**	0.306	0.373	0.454
7	**OPCA + VSM**	0.328	0.395	0.473
8	**S2Net**	0.329	0.401	0.479
9	**S2Net + VSM**	0.340	0.407	0.483

Table 2: Web document ranking results using different linear projection models, tested on the evaluation data set, where only the title field of each document is used.

OPCA (Rows 6 and 7) leverages the pair information in a more principled way [12, 27] than CL-LSA does. Noticing that solving the Eigen-decomposition problem in PCA is the same as finding the vectors $\boldsymbol{v}$ that maximize the Rayleigh quotient:

$$\frac{\boldsymbol{v}^T \mathbf{S} \boldsymbol{v}}{\boldsymbol{v}^T \boldsymbol{v}} \tag{22}$$

OPCA improves PCA by replacing Eq. (22) with the generalized Rayleigh quotient:

$$\frac{\boldsymbol{v}^T \mathbf{S} \boldsymbol{v}}{\boldsymbol{v}^T \mathbf{N} \boldsymbol{v}} \tag{23}$$

where $\mathbf{N}$ is the noise covariance matrix. The role of $\mathbf{N}$ is to ensure that the variance of the projected vectors of the query and title from the same pair can be minimized. Let $\mathbf{C_Q}$ and $\mathbf{C_D}$ be the document-term matrices of queries and titles, respectively. In addition, column vectors in $\mathbf{C_Q}$ and $\mathbf{C_D}$ correspond respectively to the query and title in the pair when they have the same column index. The noise covariance matrix is constructed as

$$\mathbf{N} = \frac{1}{n}(\mathbf{C_Q} - \mathbf{C_D})^T (\mathbf{C_Q} - \mathbf{C_D}) \tag{24}$$

where n is the number of query-title pairs.

S2Net (Rows 8 and 9) is the learning framework introduced in Section 4.2, where the projection matrix is discriminatively learned using relevant and irrelevant pairs of queries and titles. We first randomly split the clickthrough corpus into two subsets, training (99.5%) and validation (0.5%). For each query, the paired title is treated relevant (positive) and we randomly selected 4 other titles from the data as the irrelevant ones (negative). The pairwise training setting encourages the model to lead to higher similarity scores of positive pairs compared to negative ones of the same query. We stop the training process based on the model performance on the validation set.

Several interesting conclusions can be drawn from the results shown in Table 2. First, when comparing different linear projection models with the VSM baseline (Rows 2, 4, 6 and 9 vs. Row 1), we found that all models except S2Net perform worse than VSM. This is consistent with the observation made by other researchers, which is that using LSA alone can hurt the ranking performance, especially for a very low dimensional concept vector space [24]. This result also justifies the scheme of combining the projection models with VSM. As presented in Table 2, the NDCG scores of the combined models are all better than both

VSM and the corresponding projection models. Second, unlike the case of topic models, simply extending LSA to its bilingual version CL-LSA does not lead to any significant improvement (Rows 4 and 5 vs. Rows 2 and 3). Third, by simultaneously minimizing the distance between projected vectors of queries and their paired titles, OPCA does outperform LSA and CL-LSA with a small but statistically significant margin (Row 6 vs. Rows 2 and 4). However, after combining with the term vector model, the differences among these methods are not significant (Row 7 vs. Rows 3 and 5). Finally, the S2Net-trained DPM, when either used alone or combined with the term vector model, outperforms significantly other competing models (Rows 8 and 9 vs. Rows 1 to 7). Its superior results demonstrate that with an objective tightly related to the measure of evaluating document ranking, the discriminative learning approach can be very effective.

S2Net clearly outperforms other linear projection methods, but its training process is, unfortunately, more computationally expensive. Unlike LSA, CL-LSA and OPCA, which can all be solved by Eigen-decomposition, there is no analytic solution that minimizes the loss function in S2Net. In our current implementation, using a cluster of 60 ~ 80 nodes, each training iteration takes 1 to 1.5 hours and the model converges in approximately 40 iterations. The training time scales roughly linearly in terms of the number of dimensions and the number of examples. In contrast, using a commodity 8-core server with 64-GB memory, it typically takes 8 hours or less to derive an LSA, CL-LSA or OPCA model.

6. CONCLUSION

This paper presents two new document ranking models by combining the methods of latent semantic representation and the statistical translation-based approach to IR. We explore various methods of learning the semantic representation that is shared by a query and its paired titles from clickthrough data. Our evaluation on Web search shows that the proposed clickthrough-based latent semantic models significantly outperform both the standard IR models that do not use clickthrough data and those previous clickthrough-based translation models that do not use semantic representation.

In future work, we intend to explore alternative strategies of combining latent semantic models and translation models for IR. For example, we can form query-title corpora, where both the queries and titles are labeled by topics or concepts (e.g., generated using LSA). Then we can align the corpora using word-alignment models and readily compute translation probabilities based on words and topics. Another research area is the modeling of the correlations between topics in document ranking, as suggested in [3]. This is motivated by the observation that a search user may click a document on a topic that is related to, but not the same as, the topic in her query.

7. REFERENCE

[1] Asuncion, A., Welling, M, Smyth, P., and Teh, Y W. 2009. On smoothing and inference for topic models. In *Proceedings of Uncertainty in Artificial Intelligence*, pp. 27-34.

[2] Berger, A., and Lafferty, J. 1999. Information retrieval as statistical translation. In *SIGIR*, pp. 222-229.

[3] Blei, D., and Lafferty, J. 2007. A correlated topic model of science. *The Annals of Applied Statistics*, Vol. 1, No. 1, 17-35.

[4] Blei, D. M., Ng, A. Y., and Jordan, M. J. 2003. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3: 993-1022.

[5] Brown, P. F., Della Pietra, S. A., Della Pietra, V. J., and Mercer, R. L. 1993. The mathematics of statistical machine translation: parameter estimation. *Computational Linguistics*, 19(2): 263-311.

[6] Burges, C., Shaked, T., Renshaw, E., Lazier, A., Deeds, M., Hamilton, and Hullender, G. 2005. Learning to rank using gradient descent. In *ICML*, pp. 89-96.

[7] Chien, J-T., and Wu, M-S. 2008. Adaptive Bayesian latent semantic analysis. *IEEE Trans on Audio, Speech, and Language Processing*, 16(1): 198-207.

[8] de Freitas, N., and Barnard, K. 2001. Bayesian latent semantic analysis of multimedia databases. *Tech Report TR-2001-15*, University of British Columbia.

[9] Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T., and Harshman, R. 1990. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6): 391-407

[10] Dempster, A., Laird, N., and Rubin, D. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39: 1-38.

[11] Dumais, S. T., Letsche, T. A., Littman, M. L., and Landauer, T. K. 1997. Automatic cross-linguistic information retrieval using latent semantic indexing. In *AAAI-97 Spring Symposium Series: Cross-Language Text and Speech Retrieval*.

[12] Diamantaras, K. I., and Kung, S. Y. 1996. *Principle Component Neural Networks: Theory and Applications*. Wiley-Interscience.

[13] Ganchev, K., Graca, J., Gillenwater, J., and Taskar, B. 2010. Posterior regularization for structured latent variable models. *Journal of Machine Learning Research*, 11 (2010): 2001-2049.

[14] Gao, J., He, X., and Nie, J-Y. 2010. Clickthrough-based translation models for web search: from word models to phrase models. In *CIKM*, pp. 1139-1148.

[15] Gao, J., Wu, Q., Burges, C., Svore, K., Su, Y., Khan, N., Shah, S., and Zhou, H. 2009. Model adaptation via model interpolation and boosting for web search ranking. In *EMNLP*, 505-513.

[16] Gao, J., Yuan, W., Li, X., Deng, K., and Nie, J-Y. 2009. Smoothing clickthrough data for web search ranking. In *SIGIR*.

[17] Girolami, M., and Kaban, A. 2003. On an equivalence between PLSA and LDA. In *SIGIR*, pp. 433-434.

[18] Griffiths, T. L., Tenenbaum, J. B., and Steyvers, M. 2007. Topics in semantic representation. *Psychological Review*, Vol. 114, No. 2, 211-244.

[19] Hofmann, T. 1999. Probabilistic latent semantic indexing. In *SIGIR*, pp. 50-57.

[20] Huang, J., Gao, J., Miao, J., Li, X., Wang, K., and Behr, F. 2010. Exploring web scale language models for search query processing. In *Proc. WWW 2010*, pp. 451-460.

[21] Jarvelin, K. and Kekalainen, J. 2000. IR evaluation methods for retrieving highly relevant documents. In *SIGIR*, pp. 41-48

[22] Jin, R., Hauptmann, A. G., and Zhai, C. 2002. Title language model for information retrieval. In *SIGIR*, pp. 42-48.

[23] Koehn, P., Och, F., and Marcu, D. 2003. Statistical phrase-based translation. In *HLT/NAACL*, pp. 127-133.

[24] Manning, C. D., and Schutze, H. 1999. *Foundations of Statistical Natural Language Processing*. The MIT Press.

[25] Mimno, D., Wallach, H. J., Naradowsky, J., Smith, D. A., and McCallum, A. 2009. Polylingual topic models. In *EMNLP*, pp. 880-889.

[26] Och, F. 2002. *Statistical machine translation: from single-word models to alignment templates.* PhD thesis, RWTH Aachen.

[27] Platt, J., Toutanova, K., and Yih, W. 2010. Translingual document representations from discriminative projections. In *EMNLP*, pp. 251-261.

[28] Ponte, J., and Croft, W. B. 1998. A language model approach to information retrieval. In SIGIR, pp. 275-281.

[29] Svore, K., and Burges, C. 2009. A machine learning approach for improved BM25 retrieval. In *CIKM*.

[30] Vinokourov, A., Shawe-taylor, J., and Cristianini, N. 2003. Inferring a semantic representation of text via cross-language correlation analysis. In *NIPS*, pp. 1473-1480.
[31] Wang, K., Li, X., and Gao, J. 2010. Multi-style language model for web scale information retrieval. In *SIGIR*, pp. 467-474.
[32] Wei, X., and Croft, W. B. 2006. LDA-based document models for ad-hoc retrieval. In *SIGIR*, pp. 178-185.
[33] Yih, W., Toutanova, K., Platt, J., and Meek, C. 2011. Learning discriminative projections for text similarity measures. In *CoNLL*.
[34] Zhai, C., and Lafferty, J. 2001. A study of smoothing methods for language models applied to ad hoc information retrieval. In *SIGIR*, pp. 334-342.

Appendix A: Derivation of the Projection Step in BLTM-PR Training

We derive the forms of the projected posterior probabilities in Eq. (10) and (11), and the gradient of Eq. (13). The derivation follows closely the one presented in [13], and uses the standard Lagrangian duality results.

The corresponding Lagrangian of the constrained optimization problem in Eq. (9) is

$$\max_{\lambda,\mu,\eta} \min_{\boldsymbol{P}'} L(\boldsymbol{P}', \lambda, \mu, \eta) , \tag{25}$$

where

$$\begin{aligned} L(\boldsymbol{P}', \lambda, \mu, \eta) = \mathbf{KL}(\boldsymbol{P}' || \boldsymbol{P}) \\ + \sum_t \lambda_t \left(\frac{1}{|\mathbf{q}|} \sum_{q \in \mathbf{q}} P'(z=t|q) - \frac{1}{|\mathbf{d}|} \sum_{w \in \mathbf{d}} P'(z=t|w) \right) \\ + \sum_{q \in \mathbf{q}} \mu_q \left(\sum_t P'(z=t|q) - 1 \right) \\ + \sum_{w \in \mathbf{d}} \eta_w \left(\sum_t P'(z=t|w) - 1 \right) \end{aligned}$$

The Lagrangian includes the equality constraints to ensure that we are in the desired constrained space and that we have valid distributions. A non-negativity constraint on P' can also be added, but this is not necessary as it falls out from the other conditions. The form of $P'(z=t|w)$ can be obtained by setting the derivative of $L(.)$ with respect to P' to zero as

$$\begin{aligned} &\frac{\partial L(\boldsymbol{P}', \lambda, \mu, \eta)}{\partial P'(z=t|q)} = 0 \\ &= \log P'(z=t|q) + 1 - \log P\left(z=t \middle| q, \boldsymbol{\theta}^{\mathbf{q},\mathbf{d}}\right) + \frac{\lambda_t}{|\mathbf{q}|} + \mu_q . \end{aligned}$$

Thus, we have

$$P'(z=t|q) = \frac{P\left(z=t \middle| q, \boldsymbol{\theta}^{\mathbf{q},\mathbf{d}}\right) \exp\left(-\frac{\lambda_t}{|\mathbf{q}|}\right)}{e \cdot \exp(-\mu_q)} .$$

Since $\sum_t P'(z=t|q) = 1$, we get

$$Z_{\lambda,q} = e \cdot \exp(-\mu_q) = \sum_t P\left(z=t \middle| q, \boldsymbol{\theta}^{\mathbf{q},\mathbf{d}}\right) \exp\left(-\frac{\lambda_t}{|\mathbf{q}|}\right) .$$

We end up with the following form

$$P'(z=t|q) = \frac{1}{Z_{\lambda,q}} P\left(z=t \middle| q, \boldsymbol{\theta}^{\mathbf{q},\mathbf{d}}\right) \exp\left(-\frac{\lambda_t}{|\mathbf{q}|}\right) \tag{26}$$

Similarly, we can derive the form of $P'(z=t|w)$ as

$$P'(z=t|w) = \frac{1}{Z_{\lambda,w}} P\left(z=t \middle| w, \boldsymbol{\theta}^{\mathbf{q},\mathbf{d}}\right) \exp\left(\frac{\lambda_t}{|\mathbf{d}|}\right) \tag{27}$$

where

$$Z_{\lambda,w} = \sum_t P\left(z=t \middle| w, \boldsymbol{\theta}^{\mathbf{q},\mathbf{d}}\right) \exp\left(\frac{\lambda_t}{|\mathbf{d}|}\right)$$

Notice that Eq. (26) and (27) are identical to Eq. (10) and (11), respectively. Now, we show how to estimate λ_t. Plugging Eq. (26) and (27) into Eq. (25), we have

$$\begin{aligned} L(\boldsymbol{P}', \lambda) &= \sum_{t,q} P'(z=t|q) \left(-\frac{\lambda_t}{|\mathbf{q}|} - \log Z_{\lambda,q} \right) \\ &\quad + \sum_{t,w} P'(z=t|w) \left(\frac{\lambda_t}{|\mathbf{d}|} - \log Z_{\lambda,w} \right) \\ &\quad + \sum_t \lambda_t \left(\frac{1}{|\mathbf{q}|} \sum_{q \in \mathbf{q}} P'(z=t|q) - \frac{1}{|\mathbf{d}|} \sum_{w \in \mathbf{d}} P'(z=t|w) \right) \\ &= -\sum_q \log Z_{\lambda,q} - \sum_w \log Z_{\lambda,w} \end{aligned} \tag{28}$$

We then use the gradient ascent algorithm to get the optimal λ_t

$$\lambda_t^{new} = \lambda_t^{old} + \epsilon \cdot \mathbf{g}(\lambda_t) \tag{29}$$

where ϵ is the learning rate, and the gradient $\mathbf{g}$ is computed as

$$\begin{aligned} \mathbf{g}(\lambda_t) &= \frac{\partial L(\boldsymbol{P}', \lambda)}{\partial \lambda_t} \\ &= \mathbf{E}_{\boldsymbol{P}'_{\mathbf{q}}} \left[\frac{1}{|\mathbf{q}|} \sum_{j=1}^{|\mathbf{q}|} \mathbf{1}(z_j = t) \right] - \mathbf{E}_{\boldsymbol{P}'_{\mathbf{d}}} \left[\frac{1}{|\mathbf{d}|} \sum_{j=1}^{|\mathbf{d}|} \mathbf{1}(z_j = t) \right] \end{aligned} \tag{30}$$

which is identical to Eq. (13)

Appendix B: Gradient Derivation in S2Net

We derive the gradient of the loss function in Eq. (21) as follows.

$$\mathbf{g}(\mathbf{A}) = \frac{\partial L(\Delta; \mathbf{A})}{\partial \mathbf{A}} = \frac{-\gamma}{1 + \exp(-\gamma \Delta)} \frac{\partial \Delta}{\partial \mathbf{A}} \tag{31}$$

$$\frac{\partial \Delta}{\partial \mathbf{A}} = \frac{\partial}{\partial \mathbf{A}} sim_{\mathbf{A}}(\mathbf{Q}, \mathbf{D}_1) - \frac{\partial}{\partial \mathbf{A}} sim_{\mathbf{A}}(\mathbf{Q}, \mathbf{D}_2) \tag{32}$$

$$\frac{\partial}{\partial \mathbf{A}} sim_{\mathbf{A}}(\mathbf{Q}, \mathbf{D}) = \frac{\partial}{\partial \mathbf{A}} \frac{\widehat{\mathbf{Q}}^T \widehat{\mathbf{D}}}{\left\|\widehat{\mathbf{Q}}\right\| \left\|\widehat{\mathbf{D}}\right\|} = \frac{\partial}{\partial \mathbf{A}} \frac{(\mathbf{A}^T\mathbf{Q})^T (\mathbf{A}^T\mathbf{D})}{\|\mathbf{A}^T\mathbf{Q}\| \|\mathbf{A}^T\mathbf{D}\|} \tag{33}$$

Breaking it into three parts, we have

$$\frac{\partial}{\partial \mathbf{A}} (\mathbf{A}^T\mathbf{Q})^T (\mathbf{A}^T\mathbf{D}) = \boldsymbol{Q}\widehat{\mathbf{D}}^T + \boldsymbol{D}\widehat{\mathbf{Q}}^T \tag{34}$$

$$\frac{\partial}{\partial \mathbf{A}} \frac{\mathbf{1}}{\|\mathbf{A}^T\mathbf{Q}\|} = -\left(\widehat{\mathbf{Q}}^T \widehat{\mathbf{Q}}\right)^{-\frac{3}{2}} \mathbf{Q}\widehat{\mathbf{Q}}^T \tag{35}$$

$$\frac{\partial}{\partial \mathbf{A}} \frac{\mathbf{1}}{\|\mathbf{A}^T\mathbf{D}\|} = -\left(\widehat{\mathbf{D}}^T \widehat{\mathbf{D}}\right)^{-\frac{3}{2}} \mathbf{D}\widehat{\mathbf{D}}^T \tag{36}$$

To simply the notation, let a, b, c be $\widehat{\mathbf{Q}}^T\widehat{\mathbf{D}}, 1/\left\|\widehat{\mathbf{Q}}\right\|$ and $1/\left\|\widehat{\mathbf{D}}\right\|$, respectively. Eq. (33) becomes

$$\frac{\partial}{\partial \mathbf{A}} sim_{\mathbf{A}}(\mathbf{Q}, \mathbf{D}) = -abc^3 \mathbf{D}\widehat{\mathbf{D}}^T - acb^3 \mathbf{Q}\widehat{\mathbf{Q}}^T + bc\left(\boldsymbol{Q}\widehat{\mathbf{D}}^T + \boldsymbol{D}\widehat{\mathbf{Q}}^T\right)$$