Learning Query and Document Similarities from Click-through Bipartite Graph with Metadata

Wei Wu
Microsoft Research Asia
Beijing,China,10080
wuwei@microsoft.com

Hang Li[*]
Microsoft Research Asia
Beijing, China, 100080

Jun Xu[†]
Microsoft Research Asia
Beijing, China, 100080

ABSTRACT

We consider learning query and document similarities from a click-through bipartite graph with metadata on the nodes. The metadata contains multiple types of features of queries and documents. We aim to leverage both the click-through bipartite graph and the features to learn query-document, document-document, and query-query similarities. The challenges include how to model and learn the similarity functions based on the graph data.

We propose solving the problems in a principled way. Specifically, we use two different linear mappings to project the queries and documents in two different feature spaces into the same latent space, and take the dot product in the latent space as their similarity. Query-query and document-document similarities can also be naturally defined as dot products in the latent space. We formalize the learning of similarity functions as learning of the mappings that maximize the similarities of the observed query-document pairs on the enriched click-through bipartite graph. When queries and documents have multiple types of features, the similarity function is defined as a linear combination of multiple similarity functions, each based on one type of features. We further solve the learning problem by using a new technique called Multi-view Partial Least Squares (M-PLS). The advantages include the global optimum which can be obtained through Singular Value Decomposition (SVD) and the capability of finding high quality similar queries. We conducted large scale experiments on enterprise search data and web search data. The experimental results on relevance ranking and similar query finding demonstrate that the proposed method works significantly better than the baseline methods.

Categories and Subject Descriptors

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

General Terms

Algorithms,Experimentation

[*]The author has moved to Huawei Noah's Ark Lab in Hong Kong
[†]The author has moved to Huawei Noah's Ark Lab in Hong Kong

Keywords

similarity learning, click-through, multi-view partial least squares

1. INTRODUCTION

Many tasks in Information Retrieval (IR) rely on similarities between pairs of objects. In relevance ranking, given a query, one retrieves the most relevant documents and ranks them based on their degrees of relevance to the query. The relevance between a query and a document can be viewed as a kind of similarity. In query reformulation or rewriting, queries that convey similar search intents but in different forms are created to reformulate the original query, so that the documents better meeting the users' information need can be properly retrieved and ranked. The original query and reformulated queries are in fact similar queries. In query suggestion, queries with related intents are recommended to the user, to help the user to search other useful information. Those queries are also similar queries. In all these tasks, we need to measure similarities between two objects, either query-document or query-query.

Similarity functions are usually defined based on the features of queries and documents. The relevance models in IR, including Vector Space Model (VSM) [24], BM25 [20], and Language Models for Information Retrieval (LMIR) [19, 37] can all be viewed as similarity functions between query and document feature vectors [35, 33]. Similarity between a query and a document is calculated as similarity between term frequency vectors or n-gram vectors. Similarly, queries are represented as vectors in a term space or n-gram space, and the dot product or cosine is taken as a similarity function between them [38, 36]. All the methods utilize features to calculate similarity functions and we call them feature based methods.

Recently, mining query and document similarities from a click-through bipartite graph has been proposed (cf., [7, 18]). The click-through bipartite graph, which represents users' implicit judgments on query-query, document-document, and query-document relevance relations, has been proved to be a valuable source for measuring the similarities. For example, queries which share many co-clicked documents may represent similar search intents, and they can be viewed as similar queries [4, 31][1]. These methods only rely on the structure of a bipartite graph. We call them graph based methods.

In this paper, we consider leveraging information from both a click-through bipartite graph and features to measure query and document similarities. In other words, this is about how to combine the feature based methods and graph based methods. As far as we know, this problem was not well studied previously. We formalize the issue as that of learning query and document similarities from a click-through bipartite graph with metadata on the

[1]In this paper query similarity and document similarity are defined from the viewpoint search intents.

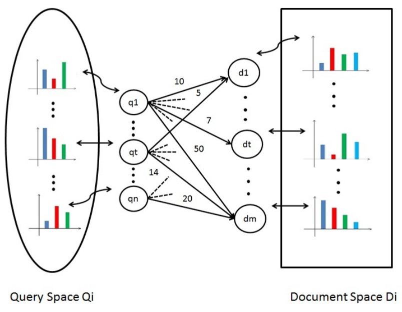

Figure 1: Click-through bipartite graph with metadata on nodes, representing queries and documents in feature spaces and their associations.

nodes representing multiple types of features. The features may come from multiple sources. For example, for queries, the content, the semantic classes, and the user information can be defined as features; for documents, features can be extracted from the URLs, titles, bodies, and anchor texts. Formally, we assume that there is a query-document bipartite graph. The bipartite graph consists of triplets (q,d,t), where q denotes a query, d denotes a document, and t denotes the number of clicks between q and d. Besides, we assume that the queries and documents on the graph have multiple types of features. For each type i ($i \geqslant 1$), queries and documents are represented as vectors in the query space $\mathcal{Q}_i$ and the document space $\mathcal{D}_i$. $\mathcal{Q}_i$ and $\mathcal{D}_i$ are subspaces of the Euclidian spaces $\mathbb{R}^{s_{qi}}$ and $\mathbb{R}^{s_{di}}$, where s_{qi} and s_{di} represent the dimensionalities of the Euclidean spaces. $\mathcal{Q}_i$ and $\mathcal{D}_i$ may or may not be the same space, which means that queries and documents may be either homogeneous or heterogeneous data. Figure 1 illustrates the relationships.

Our goal is to accurately and efficiently learn the similarity functions from the enriched click-through bipartite graph. There are several challenges: 1) how to model the similarity functions based on the complicated graph data, particularly when there are multiple types of features; 2) how to accurately learn the similarity functions; 3) how to efficiently perform the learning task. We propose a method that solves all the problems in a theoretically sound way.

1) Specifically, for each type of features, we use two linear mappings to project the query vectors in the query space and the document vectors in the document space into the same latent space. We take the dot product of the images in the latent space as the similarity function between the query and document vectors. Figure 2 illustrates the method. The dot product in the latent space is also taken as the similarity function between query and query and the similarity function between document and document. The two mappings are supposed to be different, because of the difference between the query space and document space. The final similarity functions are defined as linear combinations of similarity functions from different feature types.

2) We learn the mappings and combination weights using the enriched click-through bipartite graph data. The number of clicks between queries and documents indicates similarities between queries and documents. We formalize the learning method as an optimization problem, in which the objective is to maximize the similarities of the observed query-document pairs on the click-through bipartite graph. We regularize the combination weights with ℓ_2 norm and we make orthogonal assumptions on the mappings. We propose a new machine learning technique called Multi-view Partial Least Squares (M-PLS) to learn the linear mappings. M-PLS extends the Partial Least Squares (PLS) technique to a multi-view learning method (multiple feature types). When there is only one type of features (one view), M-PLS degenerates to the conventional PLS. Moreover, if there is no metadata used, then M-PLS becomes equivalent to Latent Semantic Indexing (LSI) and thus our method can also be regarded as an extension of LSI. We prove that although the optimization problem is not convex, the globally optimal solution of M-PLS can be easily obtained. We also conduct theoretical analysis on the method, and point out that it has two properties that enable the method to capture query-query similarity well (this is also true for document-document similarity), although the learning of it is not explicitly incorporated into the formulation.

3) The learning task can be efficiently performed through Singular Value Decomposition (SVD). First, we employ the power method to build an SVD solver. After that, we solve a simple quadratic program to learn the optimal combination weights. We show that the quadratic program has a closed-form solution.

The learned similarity functions can be applied to relevance ranking and similar query finding. In relevance ranking, the query-document similarity function can be directly used as a model for retrieval and ranking. In addition, it can also be used as a feature of a learning to rank model. In similar query finding, the similar queries found with the query-query similarity function can be utilized in tasks such as query suggestion and query reformulation.

We conducted experiments on large scale enterprise search data and web search data. The results on relevance ranking and similar query finding show that our method significantly outperforms the baseline methods. Specifically, in relevance ranking, we compare our method with the state of the art feature based methods such as BM25, graph based methods such as random walk and their linear combinations. Our method not only significantly outperforms the feature based methods and graph based methods, but also significantly performs better than their linear combinations. In similar query finding, we use examples to demonstrate the capability of our method on finding high quality similar queries. When compared with the baseline methods such as random walk and cosine similarity, our method significantly outperforms the feature based methods and graph based methods, and is comparable with the best linear combination. Particularly, we find our method performs better than the baselines on tail queries, and the results also strongly support the conclusion of our theoretical analysis.

Our contributions in this paper include: 1) proposal of learning query and document similarities from an enriched click-through bipartite graph; 2) proposal of a new learning method called M-PLS to perform the learning task; 3) theoretical proof of the properties of the proposed method and empirical demonstration of the effectiveness of the method.

The rest of the paper is organized as follows: in Section 2, we conduct a survey on related work. Then, we define the similarity learning problem in Section 3. After that, we explain our method based on M-PLS in Section 4. Experimental results are reported in Section 5, and Section 6 concludes the paper and provides some future research directions.

2. RELATED WORK

Partial Least Squares (PLS) [21] refers to a class of algorithms in statistics that aims to model the relations between two or more sets of data by projecting them into a latent space. The underlying idea of PLS algorithms is to model collinearity between different data sets. Since the first work by Wold [32], many variants of PLS have been developed and applied into many tasks such as regression [27], classification [3] and dimension reduction [26]. In practice, PLS has been successfully applied in chemometrics [21], biometrics [6], computer vision [26] and graph mining [23]. In this paper, we extend PLS to M-PLS and employ M-PLS in web search. If

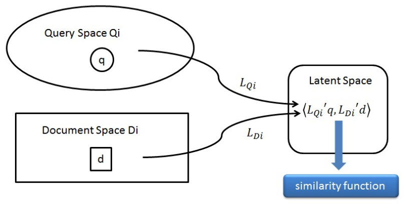

Figure 2: Projecting queries and documents from the query space and document space into a latent space using two linear mappings L_{Q_i} and $L_{\mathcal{D}_i}$. The dot product in the latent space is taken as the similarity function.

there is only one type of features in the query space as well as in the document space, our similarity learning problem degenerates to a problem which can be directly solved by PLS.

Canonical Correlation Analysis (CCA) [13] or its kernelized version KCCA [11, 12] is an alternative method to PLS . Both attempt to learn linear mapping functions to project objects in two spaces into the same latent space. The difference between CCA and PLS is that CCA learns cosine as a similarity function and PLS learns dot product as a similarity function. In our work, we choose PLS instead of CCA, because it is easier to enhance the efficiency of the former. In CCA it is necessary to compute the inverse of large matrices [2], which is computationally expensive.

Measuring query and document similarities is always an important research topic in IR. Existing work on query and document similarities can be divided into two groups: feature based methods and graph based methods. In the former group, Vector Space Model (VSM) [24], BM25 [20], and Language Models for Information Retrieval (LMIR) [19, 37] make use of features, particularly, n-gram features to measure query-document similarities. As pointed out by Xu et al. [35] and others that these models can be viewed as models using the dot product between a query vector and a document vector as the query-document similarity function. Similarly, queries can also be represented as n-grams, and the cosine or dot product can be utilized as the similarity function between them [38, 36]. In [36], queries are represented as n-gram vectors, and a cosine similarity function is learned by using distance metric learning. In [5], the authors propose calculating query similarity with term and n-gram features enriched with a taxonomy of semantic classes. In [2], queries are represented as vectors in a high dimensional space with each dimension corresponding to a document. The click frequency on a document is used as the value of the corresponding dimension. In the latter group, graph based methods exploit the structure of a click-through bipartite graph to learn the query-query, document-document, and query-document similarities. For example, Latent Semantic Indexing (LSI) [9] can be employed, which uses SVD to project queries, documents and terms into a latent space, and calculates query-document, query-query, and document-document similarities through the dot product of their images in the latent space. In [15, 1], the authors propose determining the similarity of a pair of objects based on the similarity of other pairs of objects, and the final similarity measure is iteratively calculated on a bipartite graph. A click-through bipartite graph can also be used in clustering of similar queries [4, 31]. Craswell and Szummer [7] extend the idea and propose adopting a backward random walk process on a click-through bipartite graph to propagate similarity. Our method aims to leverage both features and a click-through bipartite graph to more accurately learn the similarities.

[2]`http://en.wikipedia.org/wiki/Canonical_correlation`

Learning to rank refers to supervised learning techniques for constructing ranking models using training data [16, 17]. The problem of learning to rank is different from that of similarity learning in this paper. We leverage a click-through bipartite graph with metadata to learn similarities between queries and documents, while in learning to rank one constructs a ranking model to predict a ranking list of documents with respect to a query. The similarity learned with our method can be employed as a feature of a learning to rank model, and the similarity learning in this paper can be viewed as feature learning for learning to rank.

Methods for learning similarities between objects by utilizing bipartite graphs built from multiple sources have also been studied. In [8], term relations and document relations are integrated into a document-term bipartite graph. In [29], the authors extend LSI when information from different types of bipartite graphs is available. In [34], the authors use a unified relationship matrix to represent different types of objects and their relations. In [18], matrix factorization is simultaneously conducted on two bipartite graphs, a user-query bipartite graph and a query-document bipartite graph. In [10], similarity scores are first calculated by the LMIR model and then the scores are propagated on a click-through bipartite graph to find similar queries. The method proposed in this paper is different from these existing methods. In our method, we make use of both a click-through bipartite graph and multiple types of features on the nodes. In that sense, we combine feature based methods and graph based methods. In contrast, the existing methods are all graph-based methods and they use either more than one bipartite graph or a graph with more than one type of relations.

The method proposed in this paper is also related to multi-view learning [22] in machine learning. In multi-view learning, instances are assumed to have multiple types of features and the goal is to exploit the different types of features to learn a model. In our work, we assume that queries and documents on a click-through bipartite graph have multiple types of features, which is similar to the assumption in multi-view learning. Our work is unique in that we perform multi-view similarity learning on an enriched click-through bipartite graph. Our method of Multi-view PLS can be regarded as an extension of PLS to multi-view learning.

The similarity learning problem studied in this paper is not limited to search, and it can be potentially extended to other applications such as recommendation system and online advertisement. Recently, Wang et al. [28] propose a method for advertisement in a setting similar to ours. We learn query and document similarities from a bipartite graph with metadata, while they predict possible clicks between users and advertisements on a network with metadata. Note that there is significant difference between our method and their method. First, they only measure similarity between users and advertisements (corresponding to queries and documents), while we also measure query-query and document-document similarities. Second, their method assumes that the latent space is predefined and is independent from the similarity function, while in our method the latent space as well as its similarity function are learned from data.

3. PROBLEM FORMULATION

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ denote a click-through bipartite graph. $\mathcal{V} = Q \bigcup D$ is the set of vertices, which consists of a set of query vertices $Q = \{q_i\}_{i=1}^m$ and a set of document vertices $D = \{d_i\}_{i=1}^n$. $\mathcal{E}$ is the set of edges between query vertices and document vertices. The edge $e_{ij} \in \mathcal{E}$ between query vertex q_i and document vertex d_j is weighted by the click number t_{ij}. We assume that $\mathcal{G}$ is an undirected graph. Besides, we assume that there exists rich metadata on the vertices of the graph. The metadata consists of l types of features ($l \geqslant 1$). The features may stand for the content of queries and documents and the clicks of queries and documents on the bipartite graph [2], as will be seen in Section 5. For each type i ($1 \leqslant i \leqslant l$), query $q \in Q$ and document $d \in D$ are represented as vectors in space $\mathcal{Q}_i$ and space $\mathcal{D}_i$, respectively, where $\mathcal{Q}_i$ and $\mathcal{D}_i$ are subspaces of the Euclidean spaces $\mathbb{R}^{s_{qi}}$ and $\mathbb{R}^{s_{di}}$. Figure 1 illustrates the relationships.

Our goal is to leverage both the click-through bipartite graph and the features to learn query and document similarities. The similarities may include query-document similarity, query-query similarity, and document-document similarity. In this paper we only study query-document similarity and query-query similarity. The same method can be applied to learning of document-document similarity. Formally, we learn two similarity functions $f(q,d)$ and $g(q,q')$ given $\mathcal{G}$ and $\{(\mathcal{Q}_i,\mathcal{D}_i)\}_{i=1}^l$. Similarity function $f(q,d)$ measures the similarity between query $q \in Q$ and document $d \in D$, and similarity function $g(q,q')$ measures the similarity between queries $q, q' \in Q$.

The question is then how to model the similarity functions $f(q,d)$ and $g(q,q')$, and how to accurately and efficiently learn the similarity functions. We propose learning the similarity functions by linearly projecting the queries and documents in the query spaces and document spaces into latent spaces. For each type of feature space $(\mathcal{Q}_i,\mathcal{D}_i)$, we learn two linear mappings $L_{\mathcal{Q}_i}$ and $L_{\mathcal{D}_i}$. $L_{\mathcal{Q}_i}$ is an $s_{qi}\times k_i$ dimensional matrix which can map a query q from $\mathcal{Q}_i$ to the k_i dimensional latent space $\mathcal{K}_i$, and $L_{\mathcal{D}_i}$ is an $s_{di}\times k_i$ dimensional matrix which can map a document d from $\mathcal{D}_i$ to the k_i dimensional latent space $\mathcal{K}_i$. We assume that $\forall i$, $k_i \leqslant \min(s_{qi}, s_{di})$. $L_{\mathcal{Q}_i}$ and $L_{\mathcal{D}_i}$ are different mapping functions due to the difference between $\mathcal{Q}_i$ and $\mathcal{D}_i$. Given $q \in Q$ and $d \in D$, the images in the latent space $\mathcal{K}_i$ are $L_{\mathcal{Q}_i}^\top q$ and $L_{\mathcal{D}_i}^\top d$, respectively. We define similarity functions $f_i(q,d)$ and $g_i(q,q')$ as $f_i(q,d) = q^\top L_{\mathcal{Q}_i} L_{\mathcal{D}_i}^\top d$, $g_i(q,q') = q^\top L_{\mathcal{Q}_i} L_{\mathcal{Q}_i}^\top q'$. In other words, we take the dot products of images of queries and documents in the latent spaces as similarity functions. The final similarity functions $f(q,d)$ and $g(q,q')$ are defined as linear combinations of $\{f_i(q,d)\}_{i=1}^l$ and $\{g_i(q,q')\}_{i=1}^l$: $f(q,d) = \sum_{i=1}^l \alpha_i f_i(q,d)$, $g(q,q') = \sum_{i=1}^l \alpha_i g_i(q,q')$, where $\alpha_i \geqslant 0$ is a combination weight.

We learn the mappings $\{(L_{\mathcal{Q}_i}, L_{\mathcal{D}_i})\}_{i=1}^l$ and combination weights $\{\alpha_i\}_{i=1}^l$ from the click-through bipartite graph $\mathcal{G}$ and the features $\{(\mathcal{Q}_i,\mathcal{D}_i)\}_{i=1}^l$. Specifically, we view the click number of a query-document pair as an indicator of their similarity. We learn $\{(L_{\mathcal{Q}_i}, L_{\mathcal{D}_i})\}_{i=1}^l$ and $\{\alpha_i\}_{i=1}^l$ by maximizing the similarities of the observed query-document pairs on the click-through bipartite graph. The underlying assumption is that the higher the click number is, the more similar the query and the document are in the latent spaces.

Finally, we consider the following learning problem:

$$\underset{\{(L_{\mathcal{Q}_i},L_{\mathcal{D}_i})\}_{i=1}^l,\{\alpha_i\}_{i=1}^l}{\arg\max} \sum_{e_{uv}\in\mathcal{E}} \sum_{i=1}^l \alpha_i \cdot \left(t_{uv} \cdot q_u^{i\top} L_{\mathcal{Q}_i} L_{\mathcal{D}_i}^\top d_v^i\right), \qquad (1)$$

where q_u^i and d_v^i represent the feature vectors of query q_u and document d_v in $\mathcal{Q}_i$ and $\mathcal{D}_i$, respectively. t_{uv} represents the click number between q_u and d_v on the click-through bipartite graph $\mathcal{G}$.

Note that an alternative method for learning query and document similarities from $\mathcal{G}$ and $\{(\mathcal{Q}_i,\mathcal{D}_i)\}_{i=1}^l$ is to concatenate different types of feature vectors of queries as well as different types of feature vectors of documents and learn two mappings for queries and documents with the two concatenated vectors. The method is a special case of the learning method (1). We compare the performances of our proposed method (1) and this alternative method in Section 5.

Objective function (1) may go to infinity, since there are no constraints on the mappings $\{(L_{\mathcal{Q}_i}, L_{\mathcal{D}_i})\}_{i=1}^l$ and the weights $\{\alpha_i\}_{i=1}^l$. The features $\{(q_u^i, d_v^i) \mid e_{uv} \in \mathcal{E}, 1 \leqslant i \leqslant l\}$ are also not bounded. We consider adding proper constraints to the mapping functions $\{(L_{\mathcal{Q}_i}, L_{\mathcal{D}_i})\}_{i=1}^l$ and the weights $\{\alpha_i\}_{i=1}^l$, as shown in the next section.

4. MULTI-VIEW PARTIAL LEAST SQUARES

We further formalize the learning problem in (1) as a constrained optimization problem. We propose a new learning technique called Multi-view Partial Least Squares (M-PLS) to solve the problem, which is a multi-view learning version of PLS. We prove that the problem has a globally optimal solution. The optimal mappings can be obtained by Singular Value Decomposition (SVD) and the optimal weights can be obtained by quadratic programming. We present the algorithm and its time complexity. We also conduct theoretical analysis to demonstrate that our method has the capability to find high quality similar queries, although query-query similarity is not directly represented in the formulation.

4.1 Constrained Optimization Problem

First, we normalize the feature vectors such that $\forall u, v, i$, $\|q_u^i\| = 1$ and $\|d_v^i\| = 1$. Second, we add orthogonal constraints on the mapping matrices $\{(L_{\mathcal{Q}_i}, L_{\mathcal{D}_i})\}_{i=1}^l$. Finally, we introduce ℓ_2 regularization on the weights $\{\alpha_i\}_{i=1}^l$.

The similarity learning method is re-formalized as

$$\underset{\{(L_{\mathcal{Q}_i},L_{\mathcal{D}_i})\}_{i=1}^l,\{\alpha_i\}_{i=1}^l}{\arg\max} \sum_{e_{uv}\in\mathcal{E}} \sum_{i=1}^l \alpha_i \cdot \left(t_{uv} \cdot q_u^{i\top} L_{\mathcal{Q}_i} L_{\mathcal{D}_i}^\top d_v^i\right) \qquad (2)$$

$$\text{subject to} \quad L_{\mathcal{Q}_i}^\top L_{\mathcal{Q}_i} = I_{k_i\times k_i}, L_{\mathcal{D}_i}^\top L_{\mathcal{D}_i} = I_{k_i\times k_i}, \alpha_i \geqslant 0, \sum_{i=1}^l \alpha_i^2 \leqslant 1,$$

where $k_i \leqslant \min(s_{qi}, s_{di})$ is a parameter and $I_{k_i\times k_i}$ is an identity matrix. A larger k_i means preserving more information in the projection for the type of feature. Although problem (2) is not convex, we can prove that the globally optimal mappings $\{(L_{\mathcal{Q}_i}, L_{\mathcal{D}_i})\}_{i=1}^l$ can be obtained by Singular Value Decomposition (SVD), and the globally optimal weights $\{\alpha_i\}_{i=1}^l$ can be obtained by quadratic programming with a closed-form solution.

4.2 Globally Optimal Solution

Basically, there are two steps in finding the global optimum. First, for each type of features, we find the optimal mappings through solving SVD. Second, we determine the optimal combination weights.

Specifically, the objective function (2) can be re-written as

$$\begin{aligned}
&\sum_{e_{uv}\in\mathcal{E}} \sum_{i=1}^l \alpha_i \cdot \left(t_{uv} \cdot q_u^{i\top} L_{\mathcal{Q}_i} L_{\mathcal{D}_i}^\top d_v^i\right)\\
&= \sum_{i=1}^l \alpha_i \cdot \mathrm{trace}\left(\sum_{e_{uv}\in\mathcal{E}} t_{uv} \cdot q_u^{i\top} L_{\mathcal{Q}_i} L_{\mathcal{D}_i}^\top d_v^i\right)\\
&= \sum_{i=1}^l \alpha_i \cdot \mathrm{trace}\left(L_{\mathcal{D}_i}^\top \Big(\sum_{e_{uv}\in\mathcal{E}} t_{uv} d_v^i q_u^{i\top}\Big) L_{\mathcal{Q}_i}\right)\\
&= \sum_{i=1}^l \alpha_i \cdot \mathrm{trace}\left(L_{\mathcal{D}_i}^\top M_i L_{\mathcal{Q}_i}\right),
\end{aligned}$$

where M_i is defined as $\sum_{e_{uv}\in\mathcal{E}} t_{uv} d_v^i q_u^{i\top}$.

Suppose that $\forall i$, $k_i \leqslant \min(s_{qi}, s_{di})$, the following theorem indicates that the global optimum of the optimization problem

$$\underset{L_{\mathcal{Q}_i},L_{\mathcal{D}_i}}{\arg\max}\ \mathrm{trace}\left(L_{\mathcal{D}_i}^\top M_i L_{\mathcal{Q}_i}\right) \qquad (3)$$
$$\text{subject to} \quad L_{\mathcal{Q}_i}^\top L_{\mathcal{Q}_i} = I_{k_i\times k_i}, L_{\mathcal{D}_i}^\top L_{\mathcal{D}_i} = I_{k_i\times k_i}$$

can be reached through SVD of M_i:

THEOREM 4.1. *$\forall k_i \leqslant \min(s_{qi}, s_{di})$, the globally optimal solution of problem (3) exists. Furthermore, suppose that $M_i = U_i\Sigma_i V_i^\top$,* *where Σ_i is an $s_{di} \times s_{qi}$ diagonal matrix with singular values $\lambda_1^i \geqslant \lambda_2^i \geqslant \ldots \lambda_{p_i}^i \geqslant 0$, $p_i = \min(s_{qi}, s_{di})$, $U_i = (u_1^i, u_2^i, \ldots, u_{s_{di}}^i)$ where $\{u_j^i\}$ are left singular vectors, and $V_i = (v_1^i, v_2^i, \ldots, v_{s_{qi}}^i)$ where $\{v_j^i\}$ are right singular vectors. The global maximum is given by $\sum_{j=1}^{k_i} \lambda_j^i$ and the global optimum $\hat{L}_{Q_i}$ and $\hat{L}_{\mathcal{D}_i}$ are given by $\hat{L}_{Q_i} = (v_1^i, v_2^i, \ldots, v_{k_i}^i)$ and $\hat{L}_{\mathcal{D}_i} = (u_1^i, u_2^i, \ldots, u_{k_i}^i)$, respectively.*

The proof is given in Appendix. With Theorem 4.1, if we define $\Lambda_i = \sum_{j=1}^{k_i} \lambda_j^i$, then we can re-write problem (2) as

$$\begin{aligned}
&\max_{\substack{\{(L_{Q_i}, L_{\mathcal{D}_i})\}_{i=1}^l, \{\alpha_i\}_{i=1}^l \\ L_{Q_i}^\top L_{Q_i} = L_{\mathcal{D}_i}^\top L_{\mathcal{D}_i} = I_{k_i \times k_i}, \alpha_i \geqslant 0, \sum_{i=1}^l \alpha_i^2 \leqslant 1}} \sum_{e_{uv} \in \mathcal{E}} \sum_{i=1}^l \alpha_i \cdot \left(t_{uv} \cdot q_u^{i\top} L_{Q_i} L_{\mathcal{D}_i}^\top d_v^i\right) \\
= &\max_{\substack{\{\alpha_i\}_{i=1}^l \\ \alpha_i \geqslant 0, \sum_{i=1}^l \alpha_i^2 \leqslant 1}} \sum_{i=1}^l \alpha_i \max_{\substack{\{(L_{Q_i}, L_{\mathcal{D}_i})\}_{i=1}^l \\ L_{Q_i}^\top L_{Q_i} = L_{\mathcal{D}_i}^\top L_{\mathcal{D}_i} = I_{k_i \times k_i}}} \sum_{e_{uv} \in \mathcal{E}} \left(t_{uv} \cdot q_u^{i\top} L_{Q_i} L_{\mathcal{D}_i}^\top d_v^i\right) \\
= &\max_{\substack{\{\alpha_i\}_{i=1}^l \\ \alpha_i \geqslant 0, \sum_{i=1}^l \alpha_i^2 \leqslant 1}} \sum_{i=1}^l \alpha_i \Lambda_i. \qquad (4)
\end{aligned}$$

It can be proved that problem (4) has a closed-form global optimum. The optimal weights are given by

$$\hat{\alpha}_i = \frac{\Lambda_i}{\sqrt{\sum_{i=1}^l \Lambda_i^2}}, \quad 1 \leqslant i \leqslant l. \qquad (5)$$

It is easy to verify that when $l = 1$, problem (2) becomes a problem solvable by Partial Least Squares (PLS) [21, 25]. The orthogonal assumptions on the mapping matrices make it feasible to employ PLS to solve the problem. Therefore, our method of Multi-view PLS is an extension of PLS.

If we assume that only the click-through bipartite graph is used, i.e., no feature is used, then problem (2) becomes equivalent to Latent Semantic Indexing (LSI) on the click-through bipartite graph [9][3]. In other words, LSI is a special case of our method.

In problem (2), we consider using ℓ_2 norm to regularize the weights $\{\alpha_i\}_{i=1}^l$. An alternative method would be to use $\sum_{i=1}^l \alpha_i \leqslant 1$ (i.e., ℓ_1 norm) to regularize them. However, such a regularization will make the final solution become $\alpha_i = 1$, if $\Lambda_i = \max_{1 \leqslant j \leqslant l} \Lambda_j$, and $\alpha_i = 0$, otherwise. This is a degenerative solution and is not desirable. The ℓ_2 regularization in our method does not suffer from the problem.

4.3 Learning Algorithm

The learning algorithm is described in Algorithm 1. The algorithm contains two steps. First, for each type of feature, it calculates M_i, and solves SVD of M_i to learn the linear mappings. Then, it calculates the combination weights using (5).

At Step 2.*a*, it is necessary to calculate M_i. Suppose that there are n_q queries on the click-through bipartite graph $\mathcal{G}$. Each query has on average κ_q clicked documents. Then for each type of feature, the worst case time complexity of calculating M_i is of order $O(n_q \cdot \kappa_q \cdot s_{qi} \cdot s_{di})$, where s_{qi} and s_{di} denote the dimensionalities of query space Q_i and document space $\mathcal{D}_i$, respectively. Although s_{qi} and s_{di} can be very large, the query vectors and document vectors are usually very sparse. This can make the average time complexity much smaller than the worst case time complexity. Suppose that each dimension of query vectors has on average c_{qi} non-zero values, and each document vector has on average c_{di} non-zero values. The average time complexity of calculating M_i becomes of order $O(s_{qi} \cdot c_{qi} \cdot \kappa_q \cdot c_{di})$. Since c_{qi} and c_{di} are much smaller than n_q and s_{di}, M_i can be calculated very efficiently.

[3]LSI is usually used for learning the similarity between term and document from a term-document bipartite graph.

Algorithm 1

```
1: Input: click-through bipartite graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, feature spaces $\{(Q_i, \mathcal{D}_i)\}_{i=1}^l$, parameters $\{k_i\}_{i=1}^l$.
2: For each type of feature space $(Q_i, \mathcal{D}_i)$
     a. Calculate $M_i$ using $\sum_{e_{uv} \in \mathcal{E}} t_{uv} d_v^i q_u^{i\top}$.
     b. Calculate SVD of $M_i$.
     c. Take the first $k_i$ left singular vectors $(u_1^i, u_2^i, \ldots, u_{k_i}^i)$ as $\hat{L}_{\mathcal{D}_i}$, and first $k_i$ right singular vectors $(v_1^i, v_2^i, \ldots, v_{k_i}^i)$ as $\hat{L}_{Q_i}$.
     d. Calculate $\Lambda_i$ using $\sum_{j=1}^{k_i} \lambda_j^i$.
3: Calculate combination weight $\hat{\alpha}_i$ using equation (5).
4: Output: similarity functions:
     a. $\forall q, d, f(q, d) = \sum_{i=1}^l \hat{\alpha}_i \cdot q^{i\top} \hat{L}_{Q_i} \hat{L}_{\mathcal{D}_i}^\top d^i$.
     b. $\forall q, q', g(q, q') = \sum_{i=1}^l \hat{\alpha}_i \cdot q^{i\top} \hat{L}_{Q_i} \hat{L}_{Q_i}^\top q'^i$.
```

We empirically find in our experiments that sparse query and document feature vectors also make the matrix M_i sparse. In our experiments, the ratio of non-zero elements in the matrix M_i is not larger than 0.5%. Therefore, we can employ the power method (cf., [30]) to build an SVD solver. It is an iterative algorithm, and after i iterations, the i^{th} largest singular values can be obtained. Suppose that there are C non-zero elements in matrix M_i. The time complexity for calculating each singular value is $O(C + k_i \cdot \max(s_{qi}, s_{di}))$, and the total time complexity is $O(k_i \cdot C + k_i^2 \cdot \max(s_{qi}, s_{di}))$. Since M_i is sparse and C is much smaller than $s_{qi} \cdot s_{di}$, when k_i is small, SVD of M_i can be solved efficiently.

4.4 Query Similarity

Problem (2) is formalized as learning of query-document similarity. Actually the optimization can also help learning of query-query similarity (equivalently document-document similarity). Here, we show that our method is able to find high quality similar queries as well. Under the orthogonal assumptions, the results of problem (2) have two properties which can guarantee that the performance of the method on similar query finding is high as well.

For a specific feature type i, if the similarity between $q, q' \in Q_i$ is high, then the similarity between images of q and q' in the latent space is also high, provided that k_i is sufficiently large. Formally, we assume that there is a small number $\varepsilon/2 > 0$ such that $q^\top q' \geqslant 1 - \varepsilon/2$ (i.e., they have high similarity because $q^\top q' \leqslant 1$). Since $\|q\| = \|q'\| = 1$, we have $\|q - q'\|^2 = \|q\|^2 + \|q'\|^2 - 2q^\top q' \leqslant \varepsilon$. Suppose that $L_{Q_i} = (l_1^{Q_i}, \ldots l_{k_i}^{Q_i})$. We have

$$\|L_{Q_i}^\top q - L_{Q_i}^\top q'\|^2 = (q - q')^\top \sum_{j=1}^{k_i} l_j^{Q_i} l_j^{Q_i\top} (q - q') = \sum_{j=1}^{k_i} |\langle l_j^{Q_i}, q - q' \rangle|^2.$$

Since $L_{Q_i}^\top L_{Q_i} = I_{k_i \times k_i}$, we have

$$\|L_{Q_i}^\top q - L_{Q_i}^\top q'\|^2 = \sum_{j=1}^{k_i} |\langle l_j^{Q_i}, q - q' \rangle|^2 \leqslant \|q - q'\|^2 \leqslant \varepsilon.$$

Thus, we have $q^\top L_{Q_i} L_{Q_i}^\top q' \geqslant (\|L_{Q_i}^\top q\|^2 + \|L_{Q_i}^\top q'\|^2 - \varepsilon)/2$. This inequality indicates that for a specific feature type i, if the similarity between two queries q and q' is high in the feature space Q_i, then the similarity between their images in the latent space is determined by their norms in the space. The square of the norm of query q in the latent space can be calculated as

$$\|L_{Q_i}^{\top} q\|^2 = q^{\top} L_{Q_i} L_{Q_i}^{\top} q = q^{\top} \sum_{j=1}^{k_i} l_j^{Q_i} l_j^{Q_i \top} q = \sum_{j=1}^{k_i} |\langle l_j^{Q_i}, q \rangle|^2. \quad (6)$$

Since usually $k_i \leqslant s_{qi}$, we have $\|L_{Q_i}^{\top} q\|^2 \leqslant \|q\|^2 = 1$. With a sufficiently large k_i, we can assume that $\|L_{Q_i}^{\top} q\|^2 \geqslant 1 - \delta$, and $\|L_{Q_i}^{\top} q'\|^2 \geqslant 1 - \delta$, where $\delta \geqslant 0$ is a small number. Thus, we obtain $q^{\top} L_{Q_i} L_{Q_i}^{\top} q' \geqslant 1 - \delta - \frac{\varepsilon}{2}$. Note that δ monotonically decreases when k_i increases. In an extreme case, when $k_i = s_{qi}$, $\delta = 0$ holds. We call this property *inheritance*. This property ensures that when sufficient information is preserved in the latent space (i.e., k_i is sufficiently large), similar queries in the original feature space will also be similar in the latent space.

Second, suppose that $q_1, q_2 \in Q_i$, $d_1, d_2 \in \mathcal{D}_i$, and $q'_1 = L_{Q_i}^{\top} q_1$, $q'_2 = L_{Q_i}^{\top} q_2$, $d'_1 = L_{\mathcal{D}_i}^{\top} d_1$, and $d'_2 = L_{\mathcal{D}_i}^{\top} d_2$. If we assume that q'_1 and d'_1, q'_2 and d'_2, and d'_1 and d'_2 are similar pairs with high similarity, then when k_i is sufficiently large, we can obtain high similarity between q'_1 and q'_2. Formally, we assume that there is a small number $\varepsilon > 0$ such that $q_1'^{\top} d'_1 \geqslant 1 - \varepsilon/18$, $q_2'^{\top} d'_2 \geqslant 1 - \varepsilon/18$, and $d_1'^{\top} d'_2 \geqslant 1 - \varepsilon/18$. We have $\|q'_1 - q'_2\| \leqslant \|q'_1 - d'_1\| + \|d'_1 - d'_2\| + \|q'_2 - d'_2\|$. Since $q'_1 = L_{Q_i}^{\top} q_1$ and $d'_1 = L_{\mathcal{D}_i}^{\top} d_1$, similar to the analysis of equation (6), we have $\|q'_1\| \leqslant \|q_1\| = 1$ and $\|d'_1\| \leqslant \|d_1\| = 1$. We know that $\|q'_1 - d'_1\|^2 = \|q'_1\|^2 + \|d'_1\|^2 - 2 q_1'^{\top} d'_1 \leqslant \varepsilon/9$. Similarly, we can get $\|q'_2 - d'_2\|^2 \leqslant \varepsilon/9$ and $\|d'_1 - d'_2\|^2 \leqslant \varepsilon/9$. Thus, we obtain $\|q'_1 - q'_2\|^2 \leqslant \varepsilon$. Similarly to the analysis above, we can say that with a sufficiently large k_i, we can have the similarity between q'_1 and q'_2 large enough. We call this property *transitivity*. The property ensures that when sufficient information is preserved in the latent space, we can derive similar query pairs from similar query-document pairs. The property needs high similarity between documents in the latent space. The condition can be easily met, because document similarity can be preserved with the "inheritance" property.

In our experiments, we find that the inheritance property and the transitivity property can really help us on finding high quality similar queries. We also give an example to support our theoretical analysis.

5. EXPERIMENTS

We conducted experiments to test the performance of the proposed method on relevance ranking and similar query finding. We used two data sets: enterprise search data and web search data.

5.1 Data Sets

We collected one year click-through data from an enterprise search engine of an IT company and one week click-through data from a commercial web search engine. We built two click-through bipartite graphs with the data. If there are more than 3 clicks between a query and a document, we added a link between them. In other words we discarded the links between queries and documents whose click frequency is lower than or equal to 3. Finally, there are 51,699 queries and 81,186 documents on the bipartite graph of enterprise search data, and there are 94,022 queries and 111,631 documents on the bipartite graph of web search data. Each query has on average 2.44 clicked documents in the enterprise bipartite graph and each query has on average 1.74 clicked documents in the web bipartite graph.

We extracted features from the two click-through data sets as metadata on the bipartite graphs. In this paper, we only considered two types of metadata because our focus in this paper is verification of the effectiveness of the proposed method. First, we took the words in queries and the words in URLs and titles of documents as features, referred to as "word features". Words were stemmed and stop words were removed. With word features, each query is represented by a tf-idf vector in the query space, and each document is also represented by a tf-idf vector in the document space. Each dimension of the query space corresponds to a unique term and so does each dimension of the document space. Note that the two spaces are of high dimension and very sparse. For the enterprise search data, there are 9,958 unique terms. For the web search data, the dimensionality of term space is 101,904. Next, we followed [2] and took the numbers of clicks of documents as features of queries and the numbers of clicks of queries as features of documents. We call the features "graph features", because they are derived from the bipartite graphs. Each query is represented by a vector of click numbers in the query space and each document is represented by a vector of click numbers in the document space. Each dimension of the query space corresponds to a document on the click-through bipartite graph, and similarly each dimension of the document space corresponds to a query on the click-through bipartite graph. The dimensionalities of the two spaces are also very high, while the densities of the two spaces are very low.

In addition, we obtained relevance data consisting of judged query-document pairs from the search engine in a different time period. We also collected relevance data in the enterprise in a different time period. There are five level judgments, including "Perfect", "Excellent", "Good", "Fair", and "Bad". To make comparisons with baseline methods, we removed the queries that are not on the click-through bipartite graphs. There are 1,701 queries and associated documents having judgments in the enterprise search data. Each query has on average 16 judged documents. The number of judged queries is 4,445 in the web search data, and each query has on average 11.34 judged documents. We randomly split each data set and used half of them for tuning model parameters and the other half for model evaluation.

In summary, for both the web data and the enterprise data, we learned models on a click-through bipartite graph with metadata, tuned model parameters on some relevance data, and evaluated model performances on other relevance data.

5.2 Experiment Setup

We consider four alternatives to learn a similarity function using our method: 1) Only word features are used. We denote the model as M-PLS$_{\text{Word}}$; 2) Only graph features are used. We refer to the model as M-PLS$_{\text{Bipar}}$; 3) The vectors from the word feature space and the vectors from the graph feature space are concatenated to create long vectors. The corresponding space is the Cartesian product of the word feature space and the graph feature space. We call the model M-PLS$_{\text{Conca}}$; 4) We apply our method to learn a similarity function assuming that queries and documents have two types of features and we learn a linearly combined similarity function. We call the model M-PLS$_{\text{Com}}$.

As baselines, we consider feature based methods, graph based methods, and their linear combinations. In relevance ranking, we take BM25 as an example of feature based methods. We choose LSI on click-through bipartite graph and random walk (RW for short) [7] as examples of graph based methods. We also linearly combine LSI and RW with BM25. In similar query finding, besides LSI and RW, we adopt as baseline methods cosine similarity of two query vectors represented with graph features and cosine similarity of two query vectors represented with word features. We denote them as Cos$_{\text{B}}$ and Cos$_{\text{W}}$, respectively. We also linearly combine Cos$_{\text{B}}$, LSI, and RW with Cos$_{\text{W}}$.

Table 1: Weights in combination methods on two data sets

Model	Components	Combination weights	
		Enterprise	Web
Relevance ranking			
LSI+BM25	(LSI,BM25)	$(0.9, 0.1)$	$(0.8, 0.2)$
RW+BM25	(RW,BM25)	$(0.9, 0.1)$	$(0.8, 0.2)$
Similar query finding			
$\mathrm{Cos_B}$+$\mathrm{Cos_W}$	($\mathrm{Cos_B}$,$\mathrm{Cos_W}$)	$(0.5, 0.5)$	$(0.5, 0.5)$
LSI+$\mathrm{Cos_W}$	(LSI,$\mathrm{Cos_W}$)	$(0.5, 0.5)$	$(0.5, 0.5)$
RW+$\mathrm{Cos_W}$	(RW,$\mathrm{Cos_W}$)	$(0.5, 0.5)$	$(0.5, 0.5)$

To evaluate the performances of different methods in relevance ranking, we employ NDCG [14] at positions of 1, 3, and 5 as evaluation measures. For similar query finding, we evaluated the quality of similar queries found by each method. First, 500 queries were randomly sampled from the data for model evaluation and their top 3 most similar queries by each method were collected. We manually judged the quality of the similar queries. The final results are presented in pos-com-neg graphs, where "pos" represents the ratio of queries for which our method provides higher quality similar queries, "com" represents the ratio of queries on which the performances of our method and the baseline methods are comparable, and "neg" represents the ratio of queries on which the baseline methods do a better job.

5.3 Parameter Setting

We set the parameters of the methods in the following way. In BM25, the default setting is used. There are two parameters in random walk (RW for short): the self-transition probability and the number of transition steps. Following the conclusion in [7], we fixed the self-transition probability as 0.9 and chose the number of transition steps from $\{1, \ldots, 10\}$. We found that RW reaches a "stable" state with a few steps. In the experiments on both the web and enterprise data sets, after 5 steps we saw no improvement on the data for parameter tuning in terms of the evaluation measures. Therefore, we set 5 as the number of transition steps of RW on both data sets.

In LSI, $\mathrm{M\text{-}PLS_{Word}}$, $\mathrm{M\text{-}PLS_{Bipar}}$, $\mathrm{M\text{-}PLS_{Conca}}$, and $\mathrm{M\text{-}PLS_{Com}}$, the parameter is the dimensionality of the latent space. We set the dimensionalities in the range of $\{100, 200, \ldots, 1000\}$ for both the enterprise search data and the web search data. We found that when we increase the dimensionality of the latent space, the performances of LSI, $\mathrm{M\text{-}PLS_{Word}}$, $\mathrm{M\text{-}PLS_{Bipar}}$, $\mathrm{M\text{-}PLS_{Conca}}$, and $\mathrm{M\text{-}PLS_{Com}}$ are all improved on the data for parameter tuning. The larger the dimensionality is, the better the performance is. On the other hand, a large dimensionality means that we need to calculate more singular values and use more computation power. Therefore, we finally chose 1000 as the dimensionalities of the latent spaces for LSI, $\mathrm{M\text{-}PLS_{Word}}$, $\mathrm{M\text{-}PLS_{Bipar}}$, $\mathrm{M\text{-}PLS_{Conca}}$, and $\mathrm{M\text{-}PLS_{Com}}$ on both data sets.

In the combination models, the weights are also parameters. In LSI+BM25 and RW+BM25 for relevance ranking, LSI+$\mathrm{Cos_W}$, R-W+ $\mathrm{Cos_W}$, and $\mathrm{Cos_B}$+ $\mathrm{Cos_W}$ for similar query finding, we chose the combination weights within $\{0.1, 0.2, \ldots, 0.9\}$. For relevance ranking the weights were determined based on the results on the data for parameter tunning. For similar query finding we found that the results of LSI+$\mathrm{Cos_W}$, RW+ $\mathrm{Cos_W}$, and $\mathrm{Cos_B}$+ $\mathrm{Cos_W}$ are not sensitive to the combination weights. Therefore, we only report their results under uniform weights. Table 1 shows the weights in each combination model. Note that in $\mathrm{M\text{-}PLS_{Com}}$, the combination weights are chosen automatically using equation (5).

Table 2 shows the properties of matrices for computing SVD in each method. We can see that although the matrices have high dimensionalities, they are really sparse, and thus it is possible to conduct SVD on them efficiently.

Table 2: Properties of SVD matrices in each method

Enterprise search data		
	Dimension	Density
$\mathrm{M\text{-}PLS_{Com}}$	9958×9958, 81186×51699	0.4%, 0.08%
$\mathrm{M\text{-}PLS_{Word}}$	9958×9958	0.4%
$\mathrm{M\text{-}PLS_{Bipar}}$	81186×51699	0.08%
$\mathrm{M\text{-}PLS_{Conca}}$	91144×61657	0.5%
LSI	81186×51699	0.003%
Web search data		
	Dimension	Density
$\mathrm{M\text{-}PLS_{Com}}$	101904×101904, 111631×94022	0.008%, 0.01%
$\mathrm{M\text{-}PLS_{Word}}$	101904×101904	0.008%
$\mathrm{M\text{-}PLS_{Bipar}}$	111631×94022	0.01%
$\mathrm{M\text{-}PLS_{Conca}}$	213535×195926	0.01%
LSI	111631×94022	0.002%

Table 3: Relevance ranking result on enterprise search data

	NDCG@1	NDCG@3	NDCG@5
$\mathrm{M\text{-}PLS_{Com}}$	**0.715**	**0.733**	**0.747**
$\mathrm{M\text{-}PLS_{Conca}}$	0.700	0.728	0.742
$\mathrm{M\text{-}PLS_{Word}}$	0.688	0.718	0.739
$\mathrm{M\text{-}PLS_{Bipar}}$	0.659	0.684	0.705
BM25	0.653	0.657	0.663
RW	0.654	0.683	0.700
RW+BM25	0.664	0.688	0.705
LSI	0.656	0.676	0.695
LSI+BM25	0.692	0.701	0.712

5.4 Experimental Results

5.4.1 *Relevance Ranking Results*

Table 3 and Table 4 give the evaluation results on relevance ranking for the two data sets. We can see that on both data sets, our method $\mathrm{M\text{-}PLS_{Com}}$ not only outperforms the state of the art feature based methods such as BM25 and graph based methods such as RW, but also performs better than their linear combinations. We conducted sign test on the improvements of $\mathrm{M\text{-}PLS_{Com}}$ over the baseline methods. The results show that all the improvements are statistically significant ($p < 0.01$).

$\mathrm{M\text{-}PLS_{Conca}}$ also performs well on both data sets. The higher complexity of it makes it less attractive than $\mathrm{M\text{-}PLS_{Com}}$ (Note that the matrix of $\mathrm{M\text{-}PLS_{Conca}}$ for SVD has a high dimensionality and is dense). Therefore it is better to conduct multi-view PLS instead of single view PLS. $\mathrm{M\text{-}PLS_{Bipar}}$ performs worst among the alternatives of our method. This may be because 1) the features of $\mathrm{M\text{-}PLS_{Bipar}}$ are more sparse (cf., Table 2); 2) both the features and the similarities (i.e., click numbers) in learning of $\mathrm{M\text{-}PLS_{Bipar}}$ are from the click-through bipartite graph and there is overlap between them.

More interestingly, we find that when we linearly combine our method $\mathrm{M\text{-}PLS_{Com}}$ with BM25, the performances of our method can be further improved, indicating that our method $\mathrm{M\text{-}PLS_{Com}}$ is complementary to BM25. Table 5 shows the results. Note that the learned linear combination model can be viewed as a simplified learning to rank model.

5.4.2 *Similar Query Finding Results*

We show the performance of the proposed method on similar query finding. We compared $\mathrm{M\text{-}PLS_{Com}}$ with other baseline methods on the two data sets. In each data set, we evaluated the quality

Table 4: Relevance ranking result on web search data

	NDCG@1	NDCG@3	NDCG@5
M-PLS$_{\text{Com}}$	**0.681**	**0.731**	**0.739**
M-PLS$_{\text{Conca}}$	0.676	0.728	0.736
M-PLS$_{\text{Word}}$	0.674	0.726	0.732
M-PLS$_{\text{Bipar}}$	0.612	0.680	0.693
BM25	0.637	0.690	0.690
RW	0.655	0.704	0.704
RW+BM25	0.671	0.718	0.716
LSI	0.588	0.665	0.676
LSI+BM25	0.649	0.705	0.706

Table 5: Performances of combination of M-PLS$_{\text{Com}}$ and BM25 on relevance ranking

	NDCG@1	NDCG@3	NDCG@5
Enterprise search data	0.727	0.737	0.749
Web search data	0.697	0.740	0.745

of similar queries of 500 random queries found by each method. Finally, we presented the comparisons through pos-com-neg graphs.

Figure 3 and Figure 4 show the results. We can see that M-PLS$_{\text{Com}}$ significantly outperforms the feature based methods and graph based methods, including Cos$_{\text{B}}$, Cos$_{\text{W}}$, LSI and RW. Our method performs better on more than 15% queries from the enterprise data and on more than 25% queries from the web data. Only on less than 7% queries from the enterprise data and less than 3% queries from the web data, our method performs worse. We conducted sign test, and the results show that all the improvements are statistically significant ($p < 0.01$). Among the combined models, Cos$_{\text{B}}$ + Cos$_{\text{W}}$ and RW + Cos$_{\text{W}}$ perform comparably well, and our method still outperforms them. However, the improvement of our method over those methods is not so significant.

We investigated the reasons that the methods can or cannot achieve high performance in similar query finding. First, we found that Cos$_{\text{B}}$, LSI and RW are good at handling head queries, since they all rely on co-clicks on the click-through bipartite graph to calculate query similarity. Among them, RW performs a little better than Cos$_{\text{B}}$. This may be because similarity on the bipartite graph can be propagated by RW. In contrast, for tail queries, we cannot expect these methods to have good performances. In an extreme case, if a query only has one clicked document and the document is only clicked by the query, then no similar queries can be found for the query. We call this kind of query "isolated island". Second, no matter whether two queries have co-clicked documents, if they share some words, they are likely to be judged as similar queries by Cos$_{\text{W}}$. Therefore, Cos$_{\text{W}}$ is good at handling queries sharing terms. This is especially true for tail queries, since tail queries tend to be longer. However, if two similar queries do not share any term, their similarity cannot be captured by Cos$_{\text{W}}$. The problem is called "term mismatch", and becomes serious when the queries have spelling errors, abbreviations, and concatenations. The two types of methods (either use click graph or use terms) are complementary, and thus when combined together, it is possible to find similar queries with higher probability.

We found that on most queries our method M-PLS$_{\text{Com}}$ performs equally well with the combination baseline Cos$_{\text{B}}$ + Cos$_{\text{W}}$. Sometimes, the two methods even give the same top 3 most similar queries. It indicates that click-through bipartite graph and features are really useful for similar query finding. On the other hand, we also found that our method can work very well for some really difficult queries on which graph based methods and word based methods fail. The result demonstrates that our method really has the capability to leverage the enriched click-through bipartite graph to learn query similarities.

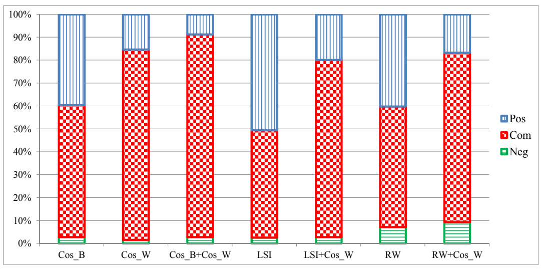

Figure 3: Similar query evaluation on enterprise search data.

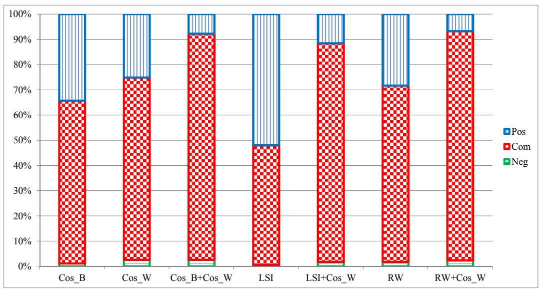

Figure 4: Similar query evaluation on web search data.

Table 6 shows some examples on the similar queries found by M-PLS$_{\text{Com}}$. We can see that M-PLS$_{\text{Com}}$ is able to find high quality similar queries, even for some "difficult" queries, which contain typos, abbreviations, concatenations and rare words. For the tail queries consisting of rare words, usually it is very hard for the baseline methods to find their similar queries, because they only have a few clicks. Nonetheless, M-PLS$_{\text{Com}}$ is still able to find similar queries for them, which is really surprising.

Finally, we particularly investigated the performance of our method on tail queries on the web data, since the web data set is larger and more sparse than the enterprise data set. We treat queries with total click numbers less than 10 on the click-through bipartite graph as tail queries. There are totally 212 tail queries in the 500 samples from the web data. Figure 5 gives the "pos-com-neg" comparison results. We can see that on tail queries, our method M-PLS$_{\text{Com}}$ performs even better than itself on the whole sample set. We further investigated the reasons that our method is capable of finding high quality similar queries in the tail. Table 7 shows an example. The query 'walmartmoneycard' is a concatenation, and is also an isolated island. As a result, for Cos$_{\text{B}}$, LSI and RW, no similar queries can be found. Cos$_{\text{W}}$ suffers from term mismatch, and thus only those queries that can exactly match 'walmartmoneycard' are returned. The combination methods can only use Cos$_{\text{W}}$, and thus return the same results with Cos$_{\text{W}}$. These baseline methods cannot find similar queries for 'walmartmoneycard'. In contrast, M-PLS$_{\text{Com}}$ can work better than the baseline methods. We found that the key reason is that our method can effectively use the click-through bipartite graph, specifically similar documents on the graph. The query 'walmartmoneycard' has a clicked document `https://www.walmartmoneycard.com/`, and its similar query 'wal-mart prepaid visa activation' also has a clicked document `https://www.walmartmoneycard.com/walmart/homepage.aspx`. Through the optimization, both the similarity between 'wal-

Table 6: Examples of similar queries found by M-PLS$_{\text{Com}}$

Original Query	Similar Queries
Spelling Errors	
dictonary	web dictionary
	onlinedictionary
	dictionary
wickapedia	www.wikipedia.org
	wikepedia
	www.wikipedia.com
gooole.com	wwww.google.com
	www.gooogle
	www.goolgle.com
Abbreviations	
fl.lottery	lottery florida
	florida lottery numbers
	florida lottery results
ym	yahoo messanger
	yahoomessenger
	yahoo im messenger
bofa online banking	bank america online banking
	bank of america online
	www.bank of america online banking
Concatenations	
dickssportinggoods	dicks sporting goods
	dicks sporting good store
	dicks sporting goods coupons
googlenews	google news
	goggle news
	news.google.com
peoplesearch	search people
	people search
	yahoo people search
Tail Queries	
star wars anniversary edition lego darth vader fighter	www.star wars legos.com
	star wars legos
	star wars lego
american express online account summary	american express account online
	american express account
	american express online
read full books online free	read books online free
	free online books to read
	read books online

Table 7: Result on a difficult query

Query: walmartmoneycard	
M-PLS$_{\text{Com}}$	www.walmartmoneycard.com
	walmartmoneycard.com
	wal-mart prepaid visa activation
Cos$_B$, LSI, RW	N / A
	N / A
	N / A
Cos$_W$	www.walmartmoneycard.com
	walmartmoneycard.com
	N / A
Cos$_B$ + Cos$_W$, LSI + Cos$_W$, RW+ Cos$_W$	www.walmartmoneycard.com
	walmartmoneycard.com
	N / A

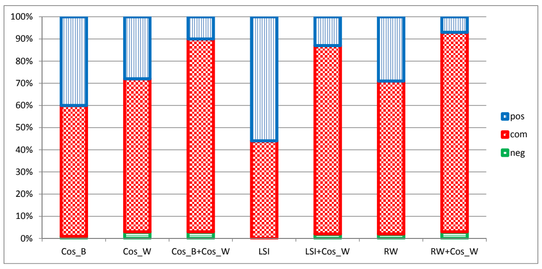

Figure 5: Similar query evaluation on the tail queries from web search data.

martmoneycard' and `https://www.walmartmoneycard.com/` and the similarity between 'wal-mart prepaid visa activation' and `https://www.walmartmoneycard.com/walmart/homepage.aspx` are maximized in the latent space (they are 0.71 and 0.53, respectively). Moreover, since the two documents share a common term 'walmartmoneycard' in the term space, their similarity is also captured in the latent space through the learning process of our method (with similarity 0.47). Therefore, with the two similar documents as a bridge, our method can connect the two queries and treat them as similar queries.

6. CONCLUSION AND FUTURE WORK

In this paper, we have studied the issue of learning query and document similarities from a click-through bipartite with metadata. The click-through bipartite represents the click relations between queries and documents, while the metadata represents multiple types of features of queries and documents. We aim to leverage both the click-through bipartite and features to perform the learning task. We have proposed a method that can solve the problem in a principled way. Specifically, for each type of features, we use two different linear mappings to project queries and documents into a latent space. Then we take the dot product in the latent space as the similarity function between query-document pairs. Similarities between query-query and document-document pairs are simultaneously defined. The final similarity function is defined as a linear combination of similarity functions from different types of features. We learn the mappings and combination weights by maximizing the similarities of the observed query-document pairs on the click-through bipartite graph, and make orthogonal assumptions on the mappings and regularize the weights using ℓ_2 norm. We have proved that the learning problem has a global optimum. The mappings can be obtained efficiently through Singular Value Decomposition (SVD) and the weights can be obtained from a closed-form solution of a quadratic program. It turns out to be a new learning method as an extension of Partial Least Squares, referred to as Multi-view PLS. We have theoretically analyzed the proposed method, and demonstrated its capability on finding high quality similar queries (also similar documents). We have conducted experiments on large scale enterprise search and web search data to test the performance of our method. The results not only indicate that our method can significantly outperform the state of the art methods on relevance ranking and similar query finding, but also verify the correctness of our theoretical analysis.

As future work, we want to further enhance the efficiency of our method and test its performance on larger data sets. To achieve the goal, we may need to parallelize the learning algorithm. We want to incorporate the similarity learned by our method into a large learning to rank system and investigate whether the similarity can improve the performance of the learning to rank system. We also want to study the kernelization of our method, so that our method is still applicable when kernel matrices instead of features are available.

7. REFERENCES

[1] I. Antonellis, H.G. Molina, and C.C. Chang. Simrank++: Query rewriting through link analysis of the click graph. *VLDB*, 1(1):408–421, 2008.

[2] R. Baeza-Yates and A. Tiberi. Extracting semantic relations from query logs. In *SIGKDD*, pages 76–85, 2007.

[3] M. Barker and W. Rayens. Partial least squares for discrimination. *Journal of chemometrics*, 17(3):166–173, 2003.

[4] D. Beeferman and A. L. Berger. Agglomerative clustering of a search engine query log. In *KDD*, pages 407–416, 2000.

[5] A. Broder, P. Ciccolo, E. Gabrilovich, V. Josifovski, D. Metzler, L. Riedel, and J. Yuan. Online expansion of rare queries for sponsored search. In *WWW'09*, pages 511–520, 2009.

[6] H. Chun and S. Kelecs. Sparse partial least squares regression for simultaneous dimension reduction and variable selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(1):3–25, 2010.

[7] N. Craswell and M. Szummer. Random walks on the click graph. In *SIGIR*, pages 239–246, 2007.

[8] B.D. Davison. Toward a unification of text and link analysis. In *SIGIR'03*, 2003.

[9] S. Deerwester, S. Dumais, G. Furnas, T. Landauer, and R. Harshman. Indexing by latent semantic analysis. *JASIS*, 41:391–407, 1990.

[10] H. Deng, M.R. Lyu, and I. King. A generalized co-hits algorithm and its application to bipartite graphs. In *KDD'09*, 2009.

[11] D.R. Hardoon and J. Shawe-Taylor. Kcca for different level precision in content-based image retrieval. In *Proceedings of Third International Workshop on Content-Based Multimedia Indexing, IRISA, Rennes, France*, 2003.

[12] D.R. Hardoon and J. Shawe-Taylor. Sparse canonical correlation analysis. *stat*, 1050:19, 2009.

[13] D.R. Hardoon, S. Szedmak, and J. Shawe-Taylor. Canonical correlation analysis: An overview with application to learning methods. *Neural Computation*, 16(12):2639–2664, 2004.

[14] K. Jarvelin and J. Kekalainen. Ir evaluation methods for retrieving highly relevant documents. In *SIGIR' 00*, pages 41–48, 2000.

[15] G. Jeh and J. Widom. Simrank: a measure of structural-context similarity. In *SIGKDD*, pages 538–543, 2002.

[16] H. Li. Learning to rank for information retrieval and natural language processing. *Synthesis Lectures on Human Language Technologies*, 4(1):1–113, 2011.

[17] T.Y. Liu. Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval*, 3(3):225–331, 2009.

[18] H. Ma, H.X. Yang, I. King, and M. R. Lyu. Learning latent semantic relations from clickthrough data for query suggestion. In *CIKM*, pages 709–718, 2008.

[19] J.M. Ponte and W.B. Croft. A language modeling approach to information retrieval. In *SIGIR' 98*, pages 275–281, 1998.

[20] S. E. Robertson, S. Walker, S. Jones, M. Hancock-Beaulieu, and M. Gatford. Okapi at trec-3. In *TREC*, 1994.

[21] R. Rosipal and N. Krämer. Overview and recent advances in partial least squares. *Subspace, Latent Structure and Feature Selection*, pages 34–51, 2006.

[22] S. Rüping and T. Scheffer. Learning with multiple views. In *Proc. ICML Workshop on Learning with Multiple Views*, 2005.

[23] H. Saigo, N. Krämer, and K. Tsuda. Partial least squares regression for graph mining. In *SIGKDD*, pages 578–586, 2008.

[24] G. Salton and M. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc., New York, NY, USA, 1986.

[25] P.J. Schreier. A unifying discussion of correlation analysis for complex random vectors. *Signal Processing, IEEE Transactions on*, 56(4):1327–1336, 2008.

[26] W.R. Schwartz, A. Kembhavi, D. Harwood, and L.S. Davis. Human detection using partial least squares analysis. In *ICCV*, pages 24–31, 2009.

[27] R.D. Tobias. An introduction to partial least squares regression. In *Proceedings of the Twentieth Annual SAS Users Group International Conference*, pages 1250–1257, 1995.

[28] C. Wang, R. Raina, D. Fong, D. Zhou, J. Han, and G. Badros. Learning relevance from heterogeneous social network and its application in online targeting. In *SIGIR'11*, 2011.

[29] X. Wang, J.T. Sun, Z. Chen, and C.X. Zhai. Latent semantic analysis for multiple-type interrelated data objects. In *SIGIR*, pages 236–243, 2006.

[30] J.A. Wegelin. A survey of partial least squares (pls) methods, with emphasis on the two-block case. *Technical Report, No.371, Seattle: Department of Statistics, University of Washington*, 2000.

[31] J.R. Wen, J.Y. Nie, and H.J. Zhang. Query clustering using user logs. *ACM Trans. Inf. Syst.*, 20(1):59–81, 2002.

[32] H. Wold. Path models with latent variables: the nipals approach. *Quantitative sociology: International perspectives on mathematical and statistical modeling*, pages 307–357, 1975.

[33] W. Wu, J. Xu, H. Li, and O. Satoshi. Learning a robust relevance model for search using kernel methods. *JMLR*, 12:1429–1458, 2011.

[34] W. Xi, E.A. Fox, W. Fan, B. Zhang, Z. Chen, J. Yan, and D. Zhuang. Simfusion: measuring similarity using unified relationship matrix. In *SIGIR*, pages 130–137. ACM, 2005.

[35] J. Xu, H. Li, and Z.L. Zhong. Relevance ranking using kernels. In *AIRS '10*, 2010.

[36] J.F. Xu and G. Xu. Learning similarity function for rare queries. In *WSDM*, pages 615–624, 2011.

[37] C.X. Zhai and J. Lafferty. A study of smoothing methods for language models applied to information retrieval. *ACM Trans. Inf. Syst.*, 22(2):179–214, 2004.

[38] Z. Zhang and O. Nasraoui. Mining search engine query logs for query recommendation. In *WWW*, pages 1039–1040, 2006.

APPENDIX

We give the proof of Theorem 4.1 here.

PROOF. Suppose that $L_{\mathcal{Q}_i} = \left(l_1^{\mathcal{Q}_i}, \dots, l_{k_i}^{\mathcal{Q}_i}\right)$, and $L_{\mathcal{D}_i} = \left(l_1^{\mathcal{D}_i}, \dots, l_{k_i}^{\mathcal{D}_i}\right)$. Objective function (3) can be re-written as $\sum_{j=1}^{k_i} l_j^{\mathcal{D}_i\top} M_i l_j^{\mathcal{Q}_i} = \sum_{j=1}^{k_i} \langle l_j^{\mathcal{D}_i}, M_i l_j^{\mathcal{Q}_i}\rangle$. Since $M_i = \sum_{w=1}^{p_i} \lambda_w^i u_w^i v_w^{i\top}$, we have

$$\begin{aligned}
\langle l_j^{\mathcal{D}_i}, M_i l_j^{\mathcal{Q}_i}\rangle &= \langle l_j^{\mathcal{D}_i}, \sum_{w=1}^{p_i} \lambda_w^i u_w^i v_w^{i\top} l_j^{\mathcal{Q}_i}\rangle = \sum_{w=1}^{p_i} \lambda_w^i \langle u_w^{i\top} l_j^{\mathcal{D}_i}, v_w^{i\top} l_j^{\mathcal{Q}_i}\rangle \\
&\leqslant \sum_{w=1}^{p_i} \lambda_w^i |\langle u_w^i, l_j^{\mathcal{D}_i}\rangle||\langle v_w^i, l_j^{\mathcal{Q}_i}\rangle| \\
&= \lambda_{k_i}^i \sum_{w=1}^{p_i} |\langle u_w^i, l_j^{\mathcal{D}_i}\rangle||\langle v_w^i, l_j^{\mathcal{Q}_i}\rangle| + \sum_{w=1}^{k_i} (\lambda_w^i - \lambda_{k_i}^i)|\langle u_w^i, l_j^{\mathcal{D}_i}\rangle||\langle v_w^i, l_j^{\mathcal{Q}_i}\rangle| \\
&+ \sum_{w=k_i+1}^{p_i} (\lambda_w^i - \lambda_{k_i}^i)|\langle u_w^i, l_j^{\mathcal{D}_i}\rangle||\langle v_w^i, l_j^{\mathcal{Q}_i}\rangle| \\
&\leqslant \lambda_{k_i}^i \sum_{w=1}^{p_i} |\langle u_w^i, l_j^{\mathcal{Q}_i}\rangle||\langle v_w^i, l_j^{\mathcal{Q}_i}\rangle| + \sum_{w=1}^{k_i} (\lambda_w^i - \lambda_{k_i}^i)|\langle u_w^i, l_j^{\mathcal{D}_i}\rangle||\langle v_w^i, l_j^{\mathcal{Q}_i}\rangle|
\end{aligned}$$

Since $\|l_j^{\mathcal{Q}_i}\| = \|l_j^{\mathcal{D}_i}\| = 1$ and $\{u_w^i\}_{w=1}^{p_i}$ and $\{v_w^i\}_{w=1}^{p_i}$ are orthonormal, we have $\sum_{w=1}^{p_i} |\langle u_w^i, l_j^{\mathcal{D}_i}\rangle||\langle v_w^i, l_j^{\mathcal{Q}_i}\rangle| \leqslant \left[(\sum_{w=1}^{p_i}\langle u_w^i, l_j^{\mathcal{D}_i}\rangle^2)(\sum_{w=1}^{p_i}\langle v_w^i, l_j^{\mathcal{Q}_i}\rangle^2)\right]^{\frac{1}{2}} \leqslant \|l_j^{\mathcal{Q}_i}\| \cdot \|l_j^{\mathcal{D}_i}\| = 1$. Thus, we know $\langle l_j^{\mathcal{D}_i}, M_i l_j^{\mathcal{Q}_i}\rangle \leqslant \lambda_{k_i}^i + \sum_{w=1}^{k_i}(\lambda_w^i - \lambda_{k_i}^i)|\langle u_w^i, l_j^{\mathcal{D}_i}\rangle||\langle v_w^i, l_j^{\mathcal{Q}_i}\rangle|$. By taking summation on both sides, we obtain $\sum_{j=1}^{k_i}\langle l_j^{\mathcal{D}_i}, M_i l_j^{\mathcal{Q}_i}\rangle \leqslant k_i\lambda_{k_i}^i + \sum_{w=1}^{k_i}(\lambda_w^i - \lambda_{k_i}^i)(\sum_{j=1}^{k_i}|\langle u_w^i, l_j^{\mathcal{D}_i}\rangle||\langle v_w^i, l_j^{\mathcal{Q}_i}\rangle|)$. From this inequality, we obtain

$$\sum_{w=1}^{k_i} \lambda_w^i - \sum_{j=1}^{k_i}\langle l_j^{\mathcal{D}_i}, M_i l_j^{\mathcal{Q}_i}\rangle \geqslant \sum_{w=1}^{k_i}(\lambda_w^i - \lambda_{k_i}^i)(1 - \sum_{j=1}^{k_i} |\langle u_w^i, l_j^{\mathcal{D}_i}\rangle||\langle v_w^i, l_j^{\mathcal{Q}_i}\rangle|).$$

Since $L_{\mathcal{Q}_i}^\top L_{\mathcal{Q}_i} = I_{k_i\times k_i}$ and $L_{\mathcal{D}_i}^\top L_{\mathcal{D}_i} = I_{k_i\times k_i}$, we have

$$\sum_{j=1}^{k_i} |\langle u_w^i, l_j^{\mathcal{D}_i}\rangle||\langle v_w^i, l_j^{\mathcal{Q}_i}\rangle| \leqslant \left[(\sum_{j=1}^{k_i}\langle u_w^i, l_j^{\mathcal{D}_i}\rangle^2)(\sum_{j=1}^{k_i}\langle v_w^i, l_j^{\mathcal{Q}_i}\rangle^2)\right]^{\frac{1}{2}} \leqslant \|u_w^i\|\cdot\|v_w^i\| = 1.$$

Thus, $\sum_{j=1}^{k_i}\langle l_j^{\mathcal{D}_i}, M_i l_j^{\mathcal{Q}_i}\rangle \leqslant \sum_{w=1}^{k_i} \lambda_w^i$.

Particularly, letting $l_j^{\mathcal{D}_i} = u_j^i$ and $l_j^{\mathcal{Q}_i} = v_j^i$, we can obtain the global maximum. □