algorithm relies *only* on the structure of the global graph, it is likely leading to topic drift. Considering the refinement is determined by the subgraph structure, the refined score of d_1, eventually, will be the same as d_3 and d_5, while the score of d_2 will be equal to that of d_4. However, initial ranking scores provide primary information between documents and the query, which should be utilized to enhance the refinement. With the initial ranking scores, a straightforward method [32] is to combine those two parts linearly, and the result may be as follows: d_1 ranks higher than d_3, then d_5; d_2 ranks on top of d_4. We may argue that the linear combination does not make full use of the information as it treats each of them individually.

Another problem with such approaches is about the graph construction. Traditionally, the graph is either constructed by explicit link structures [25, 19], such as hyperlinks of web pages, citations of research papers, or inferred from the content information as a nearest-neighbor graph [20], (i.e., affinity graph in [32]). However, to the best of our knowledge, none of the algorithms models the graph using both content and link information. In fact the link structures, no matter explicit or implicit link information, provide insight about the properties of the documents as well as the content information. From this point of view, both content information and link structure should be taken into consideration for graph construction.

In this paper, we propose a novel unified framework to model the re-ranking algorithm, by regularizing the smoothness of ranking scores over the graph with a regularizer on the initial ranking scores. The intuition behind the model is the global consistency over the graph: *similar documents are likely to have the same ranking scores with respect to a query.* In other words, if the neighbors of a document are highly relevant to a query, this document is most likely to be relevant to the query; otherwise, if none of the neighbors of a document is relevant to the query, the document is unlikely to be relevant to the query. In addition, the refined ranking scores should be at least somewhat relevant to the initial ranking scores, which, in our framework, are constrained by a regularizer on the initial ranking scores. For instance, in Fig. 1(b), the initial ranking scores and the structure of the subgraph will be considered in a unified framework. Then the scores of d_1, d_3 and d_5 will be enhanced by each other, while those of d_2 and d_4 will be decreased. Finally, one reasonable final decreasing ranking is d_1, d_3, d_2, d_5, d_4. The re-ranking algorithm discussed in this paper is not specific to documents and may arise in other entities.

Moreover, we simultaneously incorporate the textual content with other link information in a latent space graph. To achieve this goal, we learn a latent feature x_i for each document by assuming that the content information shares the same latent space with the link information, then a latent space graph is built based on the latent feature X. The advantage of this approach is that it integrates content and link information in a single joint graph, which provides complemental information for better performance.

To illustrate our methodology, we apply the framework to literature retrieval and expert finding applications on DBLP bibliography [1] data. We compare the proposed method with the initial language model method and another PageRank-style re-ranking method. Also, we evaluate the proposed method with varying graphs. Experimental results show that the improvement in our proposed method is consistent and promising.

The main contribution of this paper is to propose a joint regularization framework to model the re-ranking algorithm, which aims to improve the precision, especially the very top returned results. *The key to refining the results is the global consistency over the graph, which leverages the graph-based model for the query-dependent ranking problems.* Another contribution of this paper is that our approach simultaneously combines the content with other link information in a latent space graph, which is validated to make an improvement compared to the traditional *tf.idf* (term frequency-inverse document frequency) space.

The rest of this paper is organized as follows. We briefly summarize related work on re-ranking model and learning latent space in Section 2. In Section 3 we present the regularization framework of our approach. Section 4 describes the details of how to build the latent space graph. Next, in Section 5, we specify the statistical language model as base ranker for information retrieval, then show a case study about expert finding. We describe the experimental setup and report a series of experiments to evaluate their effectiveness in Section 6 and Section 7. Section 8 presents our conclusions and future work.

2. RELATED WORK

With the advance of machine learning, graph-based models have been widely and successively used in information retrieval and data mining. Examples include link analysis [25, 19], semi-supervised learning [36, 33], learning to rank [28, 4, 3, 8], text classification and clustering, and document re-ranking [32, 20, 21] problems, etc.

A family of work on the structural re-ranking paradigm over a graph was proposed to refine the initial ranking scores by some generalization of PageRank [25] and HITS [19]. Kurland and Lee performed re-ranking based on centrality within graphs, through PageRank-inspired algorithm [20] and HITS-style cluster-based approach [21]. Zhang et al. [32] proposed a similar method to improve Web search results based on a linear combination of results from text search and authority ranking. However, the linear combination does not make full use of the information as it treats each of them individually.

In addition, PopRank [24] is developed to extend PageRank models to integrate heterogenous relationships between objects. Another approach suggested by Minkov et al. [23] has been used to improve an initial ranking on graph walks in entity-relation networks. Very recently, Qin et al. [28] use relational objects to enhance learning to rank with parameterized regularization models. Diaz [15] use score regularization to adjust ad-hoc retrieval scores from an initial retrieval. But those two methods do not consider multiple relationships between objects. Jin et al. [18] present a different approach for ranking refinement, which utilize the base ranker and feedbacks from users to learning a better ranking function under a supervised learning framework. We propose a different method, by regularizing the smoothness of ranking scores over the latent graph, along with a regularizer on the initial ranking scores. Our regularization framework can be viewed as a non-linear combination of both parts using an unsupervised learning method.

This work is related to existing works in machine learning, especially graph-based semi-supervised learning [36, 33, 29]. The regularization framework we proposed is closely related

to graph Laplacians [36, 33, 29], which usually assume label smoothness over the graph. Mei et al. [22] extend the graph harmonic function [36] to multiple classes. However, our work is different from theirs, as their tasks are mainly used in query-independent settings (i.e., semi-supervised classification, topic modeling), while we focus on query-dependent ranking problems.

To learn a latent space graph, this work is also related to the family of latent semantic indexing, including the basic models such as Latent Semantic Analysis (LSA) [13] and Probabilistic Latent Semantic Indexing (pLSI) [17]. The key idea of LSA [13] is to map documents to a relatively low dimensional latent space, while pLSI [17] extends LSA by adding a sounder probabilistic model. Cohn and Hofmann [11] propose pLSI+PHITS to construct the latent space by combining content with link information, using content analysis based on pLSI and link analysis based on PHITS [10]. Recently, Zhu et al. [35] propose another approach that aims to exploits both the content and link information, by using a joint factorization on the document-term matrix and the link adjacency matrix. Soon later, Zhou et al. [34] combine links with co-link patterns to learn a low-dimensional latent space by using the Laplacian on directed graph. Since the latent space has been well studied, we share the idea of the joint factorization [35] to learn the latent feature, and the difference is that we leverage the latent feature for building a latent space graph.

The concrete application, expert finding, introduced in Section 5, has been well studied based on TREC2005 and TREC2006 [9, 6, 26, 16]. Balog et al. [6] propose a query-dependent approach which directly models the knowledge of an expert from associated documents, and the statistical language model [27, 30, 31] is utilized to measure the relevance between documents with a query. In this work, we chose statistical language model as our base ranker, but our approach is performed in a real world academic field based on DBLP bibliography data, which differs from the previous methods.

3. GRAPH-BASED RE-RANKING MODEL

In this section, we propose a novel and general framework to model the re-ranking algorithm, by regularizing the smoothness of ranking scores over the graph, along with a regularizer on the initial ranking scores.

3.1 Problem Definition

Let $\mathbb{D} = (d_1, d_2, ..., d_n)$ denote the set of documents to be retrieved, where each document d_i is represented by a feature vector $x_i \in \mathbb{R}^m$. Let $G(V, E)$ be a connected graph, where nodes V corresponding to the n documents, and edges E corresponding to the explicit or implicit links between documents. We assume an $n \times n$ symmetric weight matrix W on the edges of the graph is given, which w_{ij} corresponds to the weight between d_i and d_j, and D is a diagonal matrix with entries $D_{ii} = \sum_j w_{ij}$.

Given a query q, a set of documents $\mathbb{D}_{init}$ are retrieved by a standard information retrieval model (base ranker). Generally, we are highly interested in the top returned documents. However, the base ranker tends to be imperfect. *The goal of our re-ranking model is to re-order an initially-retrieved document set $\mathbb{D}_{init}$ so as to improve precision at the very top ranks of the final results.*

3.2 Regularization Framework

Inspired by the graph-based semi-supervised methods, we propose a new framework to model the re-ranking algorithm. The intuition behind the model is the global consistency: similar documents are most likely to have similar ranking scores with respect to a query. In addition, the initial ranking scores provides invaluable information, which should also be considered in the framework. Formally, we formulate a cost function $R(F, q, G)$ in a joint regularization framework similar to [33] as follows,

$$\begin{aligned} R(F,q,G) &= \frac{1}{2}\sum_{i,j=1}^{n} w_{ij} \parallel \frac{f(d_i,q)}{\sqrt{D_{ii}}} - \frac{f(d_j,q)}{\sqrt{D_{jj}}} \parallel^2 \\ &+ \mu \sum_{i=1}^{n} \parallel f(d_i,q) - f^0(d_i,q) \parallel^2, \end{aligned} \quad (1)$$

where $\mu > 0$ is the regularization parameter, $f^0(d_i, q)$ is the initial ranking score of the document d_i with respect to query q, and $f(d_i, q)$ is the refined ranking score. Let F and F^0 denote the refined and initial ranking score vector respectively. Intuitively, the first term of the cost function defines the global consistency of the refined ranking scores over the graph, while the second term defines the constraint to fit the initial ranking scores, and the trade-off between each other can be controlled by the parameter μ.

The final ranking score vector is

$$F^* = arg \min_{F \in \mathbb{R}^{+n}} R(F, q, G). \quad (2)$$

After differentiating and simplifying [33, 36], a closed-form solution can be derived,

$$\begin{aligned} F^* &= \mu_\beta (I - \mu_\alpha S)^{-1} F^0, \\ S &= D^{-\frac{1}{2}} W D^{-\frac{1}{2}}, \\ \mu_\alpha &= \tfrac{1}{1+\mu}, and\ \mu_\beta = \tfrac{\mu}{1+\mu}, \end{aligned} \quad (3)$$

where I is an identity matrix. Note that μ_α ranges from 0 to 1, and $\mu_\alpha + \mu_\beta = 1$. In this paper, we consider the normalized Laplacian in [33], and S is symmetric and positive-semidefinite. Details about how to calculate the matrix W and S will be introduced in Section 4. Given the initial ranking scores F^0 and the matrix S, we can compute the refined ranking scores F^* directly.

For the large-scale information retrieval, the matrix S is usually very large but sparse, which can be loaded in a relatively small storage space. However, the inverse matrix $(I - \mu_\alpha S)^{-1}$ will be very dense, and may need a huge space to save it. To balance the storage space and the computation time of the inverse matrix, we suggest to approximate the Eq. 3 in a specific subgraph with a submatrix $\hat{S}$, which consists of the top-n documents according to the initial ranking scores $\hat{F}^0$. It can be found that the top ranking scores are usually far greater than the very low ranking scores. Theoretically, if the ranking scores after n are close to 0, the following approximate solution is equivalent to Eq. 3,

$$\hat{F}^* = (I - \mu_\alpha \hat{S})^{-1} \hat{F}^0. \quad (4)$$

In this equation, we eliminate the parameter μ_β as it does not change the ranking. Accordingly, it needs to calculate the inverse matrix $(I - \mu_\alpha \hat{S})^{-1}$ online. Fortunately, the matrix is usually very sparse, then the complexity time of the sparse matrix inversion can be reduced to be linear with

the number of nonzero matrix elements. In our experiments, we extract the top 5,000 documents for approximation.

To point out the difference between Eq. 1 and other similar functions in [33, 36, 22], our framework is more general as it deals with a query-dependent function $f(d_i, q)$, while other methods are mainly used in query-independent settings, including semi-supervised classification and clustering.

3.3 Connection with Other Methods

The parameter μ_α can be set from 0 to 1 to control the balance between the initial ranking scores and the global consistency over the graph. It is easy to show that if $\mu_\alpha \to 0$ ($\mu \to +\infty$), the regularization Eq. 1 puts almost all weight on the second term, and the objective function boils down to the initial base ranker *baseline*. Minimizing $R(F, q, G)$ will give us the ranking scores which best fit initial ranking scores. When $\mu_\alpha \to 1$ ($\mu \to 0$), the regularization Eq. 1 puts almost all weight on the first term, and this objective function boils down to a variation of PageRank-based model. Minimizing $R(F, q, G)$ will give us the ranking scores which best fit the global consistency over the graph.

4. LEARNING A LATENT SPACE GRAPH

With the development of the World Wide Web, documents are not isolated textual content, while they are interconnected via complex (explicit or implicit) link information. The objective of this section is to simultaneously incorporate the content with link information (or entity relational data) in a latent space graph.

In the following subsections, we first introduce the basic latent semantic indexing algorithm about how to learn a latent feature, and then introduce the joint factorization which combines the content with link information in a latent space. Finally, we briefly describe how to calculate the weight matrix W according to the feature.

4.1 Latent Semantic Analysis

As mentioned in the related work, the key idea of LSA [13] is to map documents to vector space of reduced dimensionality, the *latent semantic space*. Suppose we have a document-term matrix $C \in \mathbb{R}^{n \times m}$, it is a sparse matrix whose rows represent documents, and whose columns represent terms, where m is the number of terms. Singular vector decomposition (SVD) is performed on the matrix as follows:

$$C = U \Sigma V^T,$$

where U and V are orthogonal matrices $U^T U = V^T V = I$ and the diagonal matrix Σ contains the singular values of C. By thresholding all but the largest k singular values in Σ to zero ($= \hat{\Sigma}$), the LSA approximation of C can be obtained by $\hat{C} = U \hat{\Sigma} V^T \approx U \Sigma V^T = C$. The matrix $X = U\hat{\Sigma}$ ($X \in \mathbb{R}^{n \times k}$) defines a new representation, with each row being the k-dimensional feature vector of a document.

This can be reformulated as an optimization problem which aims to approximate matrix C by XV^T,

$$\min_{X,V} \parallel C - XV^T \parallel_F^2 + \gamma \parallel V \parallel_F^2, \tag{5}$$

where V is an $m \times k$ matrix, $\parallel . \parallel_F$ is the Frobenius norm, and γ is a small positive number. In the end, the i-th row vector of X can be thought as the latent feature vector of document d_i. Empirical studies [13] in document indexing shows that it works reasonably well when the dimension k is chosen to be 100-200 dimensions.

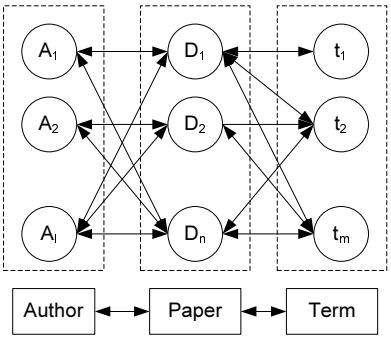


Figure 2: A sample with multiple bipartite graphs.

4.2 Embedding Content and Relational Data

Let us take the papers as an example. Besides the content, a paper is written by the authors. The relationship between papers and authors can be represented by a document-author bipartite graph, and the content of papers can be represented by a document-term bipartite graph. Fig. 2 shows a sample with multiple bipartite graphs. Moreover, a paper may be published in a conference/journal, and associated with other papers by citations. Similarly we can combine those relational data into the graph. For simplicity, we only consider the content information with the authorship information in this section. But this method can be easily extended to combine other relational data.

Now let us consider the combination of the content with the authorship information showed in Fig. 2. Given the document-author bipartite graph, it can be described by a matrix $A \in \mathbb{R}^{n \times l}$, and it is also a sparse matrix whose rows correspond to documents, and whose columns correspond to authors, where l is the number of authors. For each row of the matrix A, it means the document is authored by which authors. To combine the content and authorship information, it assumes they share the same latent space X which can approximate the content as the latent space for the authorship. Like the latent semantic indexing (LSI), we can use XV_A^T to approximate matrix A. Finally, we share the idea of joint factorization [35] to exploit them into a unified optimization problem,

$$\begin{aligned} J(X, V_C, V_A) &= \parallel C - XV_C^T \parallel_F^2 + \gamma \parallel V_C \parallel_F^2 \\ &+ \lambda \left(\parallel A - XV_A^T \parallel_F^2 + \gamma \parallel V_A \parallel_F^2 \right), \end{aligned} \tag{6}$$

where λ is the parameter for the combination, V_C is an $n \times k$ matrix, and V_A is an $l \times k$ matrix.

The joint optimization illustrated above can be solved using standard Conjugate Gradient (CG) method. The gradient for the object function J are as follows,

$$\begin{aligned} \frac{\partial J}{\partial V_C} &= \left(V_C X^T X - C^T X \right) + \gamma V_C, \\ \frac{\partial J}{\partial V_A} &= \lambda \left(V_A X^T X - A^T X + \gamma V_C \right), \\ \frac{\partial J}{\partial X} &= \left(X V_C^T V_C - C V_C \right) + \lambda \left(X V_A^T V_A - A V_A \right). \end{aligned}$$

The new representation X is ensured to capture both the

content matrix C and the authorship matrix A. In this paper, we set the parameter $\gamma = 0.1$ and $\lambda = 0.5$ based on the empirical studies for the collection.

4.3 Build Latent Space Graph

After X is calculated, we can construct the adjacency graph using K nearest neighbors (KNN). In this paper, we chose a heat kernel to define the edge weight w_{ij} as follows,

$$w_{ij} = \exp^{-\|x_i - x_j\|^2/2\sigma^2}, \tag{7}$$

where σ is a parameter for the heat kernel. So a latent space graph G is defined using the latent feature X, where the nodes denote documents and the edges E are weighted by W. After normalization, we can get the matrix $S = D^{-\frac{1}{2}} W D^{-\frac{1}{2}}$. This process is executed offline, and then we save the matrix S for our re-ranking model.

5. A CASE STUDY

To illustrate our proposed re-ranking framework, in this paper we use statistic language model as the base ranker for information retrieval, and specify the applications in literature retrieval and expert finding. In this section we introduce the statistic language model and expert finding briefly, then show the overall algorithm of our framework.

5.1 Statistic Language Model

Using language models for information retrieval has been studied extensively in recent years [27, 30, 31]. To determine the probability of a query given a document, we infer a document model θ_d for each document in a collection. With query q as input, retrieved documents are ranked based on the probability that the document's language model would generate the terms of the query, $p(q|\theta_d)$. The ranking function $f^0(d, q)$ can be written as

$$f^0(d, q) = p(q|\theta_d) = \prod_{t \in q} p(t|\theta_d)^{n(t,q)}, \tag{8}$$

where $p(t|\theta_d)$ is the maximum likelihood estimation of the term t in a document d, and $n(t, q)$ is the number of times that term t occurs in query q. The likelihood of a query q consisting of a number of terms t for a document d under a language model with Jelinek-Mercer smoothing [31] is $p(t|\theta_d) = 0.5p(t|d) + 0.5p(t)$. With the language model, we calculate the initial ranking scores of the documents with respect to a query.

5.2 Literature Retrieval and Expert Finding

Literature retrieval is a quite popular task in the academic field, and there are some well-known search engines, such as Google Scholar[1], CiteSeer[2], Libra[3], etc. Based on the literature retrieval, we address a high-level information retrieval, named as expert finding. The task of expert finding is to come up with a ranked list of experts with relevant expertise in a given query topic. For instance, given a query "data mining", it aims to find experts who have expertise in the research field "data mining", and return "Jiawei Han, etc." as the experts. This task is performed in a real world academic field based on DBLP bibliography [1], which makes it possible to recommend experts in the academic world.

[1]http://scholar.google.com/
[2]http://citeseer.ist.psu.edu/
[3]http://libra.msra.cn/

In scientific research, the publications of a researcher could be viewed as representative of his expertise [14]. This makes the assumption that authors of a paper have expertise in the topics of their papers. Although many instances exist for which well recognized researchers have published only a few manuscripts, influential researchers generally publish many manuscripts in their field. Intuitively, their expertise could thus be deduced based on the overall aggregation of their publications, which is formulated as follows:

$$P(ca, q) = \sum_{d \in \mathbb{D}_{ca}} \frac{1}{n_d} f(d, q), \tag{9}$$

where $d \in \mathbb{D}_{ca}$ means the paper d is written by the expert candidate ca, and n_d is the number of authors in the paper d. Therefore, the expert finding problem basically relies on retrieving the relevant papers. We utilize the re-ranking model to refine the ranking scores of papers, and then aggregate the refined scores to re-rank the experts.

Algorithm 1 Graph-based Re-ranking Algorithm

Offline: Learning the latent space graph
Input: $\begin{cases} C \in \mathbb{R}^{n \times m} : content\ matrix \\ A \in \mathbb{R}^{n \times l} : authorship\ matrix \end{cases}$
Perform:

1. Learn the latent feature X
   $X \leftarrow J(X, V_C, V_A)$;
2. Construct the adjacency graph W and normalize to S
   $w_{ij} = \exp^{-\|x_i - x_j\|^2/2\sigma^2}$.

Output: Return the latent space graph S.

Online: Expert finding application
Input: Given a query q,
Perform:

1. Calculate the initial ranking scores F^0 based on the language model
   $f^0(d, q) = \prod_{t \in q} p(t|\theta_d)^{n(t,q)}$;
2. Extract the top-ranked documents as a subset $\hat{\mathbb{D}}$, the corresponding ranking scores $\hat{F^0}$ and the subgraph $\hat{S}$;
3. Re-rank with the subgraph to approximate
   $\hat{F}^* = (I - \mu_\alpha \hat{S})^{-1} \hat{F^0}$;
4. Aggregate the expertise $P(ca, q)$ for authors.

Output: Return the ranked experts $\{ca_1, ca_2, ..., ca_k\}$

5.3 The Overall Algorithm

By unifying the graph-based re-ranking model and the latent space graph in Section 3 and 4, we summarize the proposed algorithm in Algorithm 1. In the algorithm, note that we first perform preprocessing (including doing tokenization, stopping and stemming) in a collection to get the content matrix C and authorship matrix A. After that, it can be divided into an online part with an offline part. In the offline algorithm, we learn the latent space X, and build the graph S. In the online algorithm, we calculate the initial ranking scores using the language model, perform re-ranking

model, and aggregate the expertise for authors.

We have implemented a research prototype search engine, named as ExpertFinding[4], to test our approach. The Lucene.Net[5] package is used for the implementation the system. To implement the re-ranking algorithm, we employ a sparse matrix package, i.e., CSparse [12], to solve the sparse matrix inversion efficiently. To deploy the efficient implementations of our scheme, all of the other algorithms used in the study are programmed in the C# language. For these experiments, the system indexes the collection and does tokenization, stopping and stemming in the usual way. We have implemented both standard *tf.idf* weighting as well the language modeling approach. The testing hardware environment is on a Windows workstation with 3.0GHz CPU and 1GB physical memory.

6. EXPERIMENTAL SETUP

In the following experiments we compare our proposed framework with other methods on the application of expert finding through an empirical evaluation. In this section we define the experimental setup, while the evaluation results are presented in Section 7.

We have defined the following task: given a query and a set of expert candidates, the system has to retrieve a list of experts who have expertise in the given area. In our evaluation we set this up as a ranking problem, i.e., the system retrieves a list of experts where the experts are ranked by the scores. In the rest of this section, we introduce the DBLP collection, the assessments and evaluation metrics.

6.1 DBLP Collection

As of November 2007, DBLP XML records contain over 955,000 articles related to Computer Science, originally published in conferences, journals, books etc., adding up to 414.5MB. Although DBLP is a good starting point for obtaining expert candidates and publications, one limitation is that each DBLP record provides the paper title without the abstract and index terms. Generally, the abstract and index terms are useful to represent the paper for estimating the probability of a query given the paper. As it is very hard to obtain the whole metadata (the abstracts and index terms of publications) for all the DBLP records, thus the Google Scholar is used for data supplement: we use the title as the query to search in Google Scholar and select the top 10 returned records which are most relevant to the query title; next, these records combined with the title are viewed as the new representation of the publication d. The metadata (HTML pages) crawled from Google Scholar is up to 20GB for all DBLP records. This process is done by a crawler and a parser automatically.

In this paper, we create our testing data set (15-CONF) from a subset of the DBLP records. We first extract all the papers published at fifteen different conferences, including KDD, SIGIR, WWW, CIKM, ICML, NIPS, IJCAI, AAAI, COLT, SIGMOD, PODS, VLDB, ICDE, CVPR, and ICCV. For each paper, we utilize the new representation (described above) of the paper d to construct the paper-term matrix C as the collection, and extract all its authors to construct the paper-author matrix A. Note that the matrix C is obtained using standard *tf.idf* weighting after doing tokenization, stopping and stemming in the usual way. The total number of papers after this process is 31437, the number of authors is 30901, and the number of terms is 16938. The statistics of the collection are shown in Table 1.

Table 1: Statistics of the 15-CONF collection

Property	#of entities
paper	31437
author	30901
term	16938

Table 2: Benchmark dataset of 16 queries

Query	#Expert
boosting	47
information retrieval	17
information extraction	20
intelligent agents	27
machine learning	42
ontology alignment	33
planning	30
privacy preservation	18
reinforcement learning	16
semantic web	44
sensor RFID data management	13
skyline	12
stream	16
support vector machine	22
semi-supervised learning	21
kernel method	22

6.2 Assessments

It is difficult to evaluate the quality of query/expert relevance rankings due to the scarcity of data that can be examined publicly. The ground truth is manually created through the method of pooled relevance judgments together with human judgments. For each query, the top authors from the computer science bibliography search engines (such as Google Scholar, CiteSeer, Libra, Rexa[6], and ArnetMiner[7]) and the committees of the top conferences in the topic were taken to construct the pool. Some researchers were then asked to assess each of the recommended candidates in context of the query. To help them in their task, those researchers were presented with publications and a description relating to each author. They could access and find additional content directly on a search engine when needed.

Such a benchmark dataset with expert lists (for expert finding) can be found in [2]. The data set contains 7 query topics and 7 expert lists. The assessments were carried out mainly in terms of how many publications he/she has published, how many publications are related to the given query, how many top conference papers he/she has published, and what distinguished awards he/she has been awarded. Four grade scores (3, 2, 1, and 0) were assigned respectively representing top expert, expert, marginal expert, and not expert.

[4]http://www.expertfinding.net/

[5]http://incubator.apache.org/lucene.net/

[6]http://rexa.info/

[7]http://www.arnetminer.org/

Finally, the judgment scores (at levels 3 and 2) were averaged to construct the final ground truth. We extended this data set to contain 16 query topics and 16 expert lists. Table 2 shows the details of the dataset.

6.3 Evaluation Metrics

For the evaluation of the task, we adopted three metrics, which capture the performance at different aspects:

Precision at rank n (P@n) Precision at rank n measures the relevance of the top n results of the retrieved list with respect to a given query topic. We report the precision P@5, P@10, P@20.

$$P@n = \frac{\#relevant\ candidates\ in\ top\ n\ results}{n}$$

Mean Average Precision (MAP) For a single query, average precision (AP) is defined as the average of the P@n values for all relevant documents.

$$AP = \frac{\sum_{n=1}^{N}(P@n * rel(n))}{R}$$

where n is the rank, N the number retrieved, and $rel(n)$ a binary function indicating the relevance of a given rank. MAP is the mean value of the average precisions computed for several queries.

Bpref Bpref [7] is the score function of the number of non-relevant candidates.

$$bpref = \frac{1}{R}\sum_{r=1}^{N}(1 - \frac{\#n\ ranked\ higher\ than\ r}{R})$$

where r is a relevant candidate and n is a member of the first R candidates judged non-relevant as retrieved by the system.

7. EVALUATION RESULTS

The presentation of the evaluation results is organized in the following three subsections. First the experiments are performed to compare the proposed method with two other methods in Section 7.1, and our method achieves the best results. Then we examine the performance of the proposed algorithm with different parameter μ_α in Section 7.2, which shows the robustness of the regularization framework. Finally, we investigate the effect of the graph construction by varying the dimensionality (k_d) and the number of nearest neighbors (k_{nn}) in Section 7.3, which shows the effectiveness of the latent feature.

7.1 Expert Finding Analysis

We consider the question whether our proposed method can boost the performance using the regularization framework on expert finding. We compare the expert finding task using our graph-based re-ranking model (*GBRM*) with two other models: one is the baseline using statistical language model, which is described in Section 5; another is a PageRank-style re-ranking model (*PRRM*) by linear combination of the global PageRank-style score and the initial ranking score [32]. We chose to compare with the *PRRM* method to see whether the linear combination of the two parts can effectively improve the performance.

Table 3: Experimental results of the baseline, PRRM and our GBRM methods (%). Best scores are in boldface

	P@5	P@10	P@20	MAP	bpref
Baseline	71.25	68.12	55.62	43.72	48.61
PRRM	73.75	65.63	51.56	39.94	44.85
GBRM	**80.00**	**73.13**	**56.87**	**45.83**	**50.70**

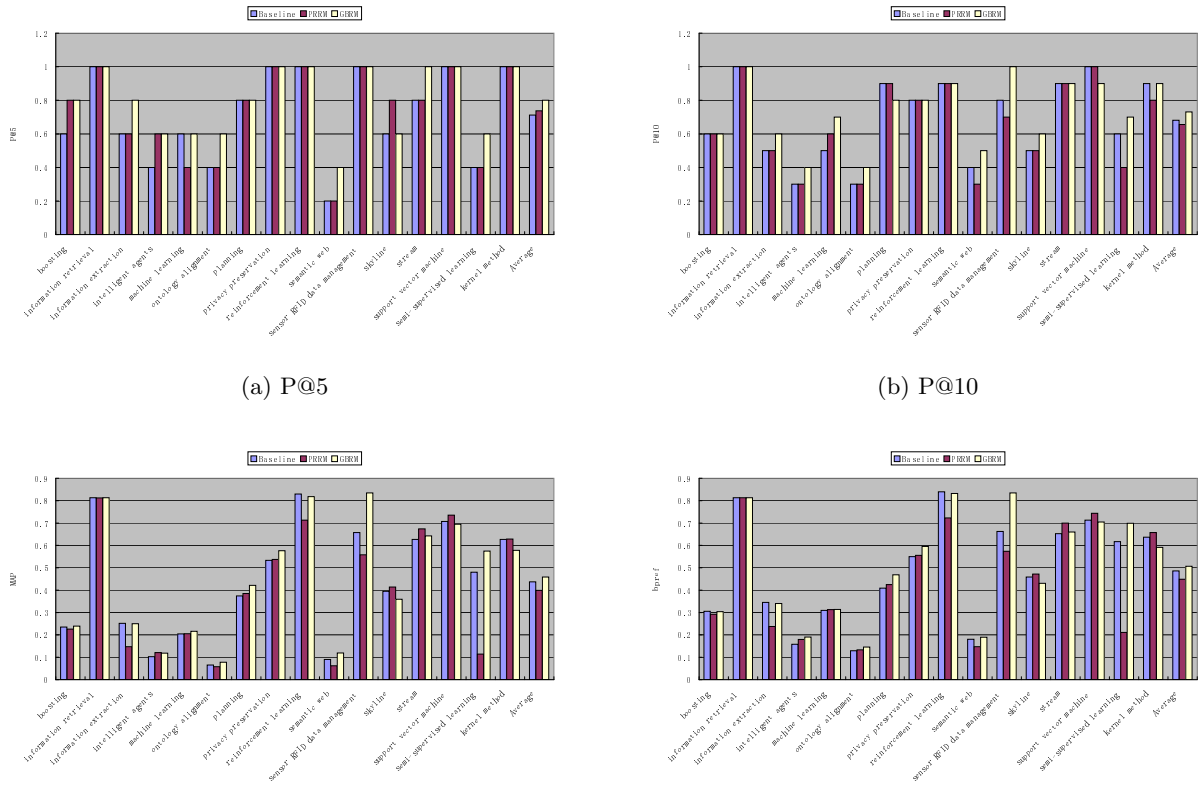
Table 3 lists the evaluation results of three different methods on the test collection. We are able to see that our proposed method outperforms both the *baseline* and *PRRM* methods in all the metrics from P@5 to bpref, especially the very top precision: for the precision P@5, our *GBRM* achieves 80%, which is about 9% higher than the *baseline* and 6% higher than the *PRRM*. For the *PRRM* method, it beats the *baseline* on the very top precision (P@5), but underperforms the *baseline* on other metrics. In addition, we notice that the performance of the *PRRM* method is degraded when more weight is put on the PageRank-style scores (currently the weight of PageRank-style scores is 0.1, while that of the initial ranking scores is 0.9). From this observation, the method to combine two parts linearly may not improve the performance since both parts are used individually. However, the improvement made in our unified regularization framework is promising.

To get an insight into the details of the results, we compare with the experimental results of the three methods on each query. Fig. 3 shows the detailed results of these methods on 16 queries. From the detailed experimental results, we can see that our *GBRM* method outperforms the *baseline* and *PRRM* method in most of the cases. On the other hand, there are few cases that our method does not make an improvement when the performance of the baseline is almost very well, for example, the query "reinforcement learning" and "support vector machine". We also found that the performance (MAP) in some queries, for instance, the query "ontology alignment" and "semantic web", is not good enough. This is because most of the data (papers/authors) corresponding to those queries are uncovered in the collection. From the figure, we can observe the improvement of our method is more consistent when compared to the *PRRM* method, which may get degradation severely (as shown in the query "semi-supervised learning") in some cases. As a result, our method achieved the best results.

7.2 Effect of Parameter μ_α

In this subsection, the effect of parameter μ_α is studied and evaluated. As mentioned in Section 3.2, the parameter μ_α is used to control the balance between the global consistency and the initial ranking scores in the unified regularization framework, and it ranges from 0 to 1. When $\mu_\alpha \to 0$, the regularization framework boils down to the initial *baseline*. If $\mu_\alpha \to 1$, the regularization framework discards the initial ranking scores, and only takes into account the global consistency on the graph according to Eq. 1.

To examine the robustness of the proposed regularization framework, we evaluate the performance of our proposed method with ten different values (from 0.1 to 0.99). Fig. 4 illustrates the experimental results for different parameter μ_α. In this figure, the solid curve denotes our *GBRM* method,



(a) P@5

(b) P@10

(c) MAP

(d) bpref

Figure 3: Experimental results of three methods on each query.

Table 4: Experimental results with different dimensionality (%). Best scores are in boldface

	P@5	P@10	P@20	MAP	bpref
TFIDF	77.50	70.63	57.50	44.70	49.52
$k_d = 20$	72.50	71.88	**58.13**	45.36	50.44
$k_d = 50$	77.50	71.88	57.50	45.77	**50.86**
$k_d = 100$	**80.00**	**73.13**	56.88	**45.84**	50.70

Table 5: Experimental results with different number of nearest neighbors (%)

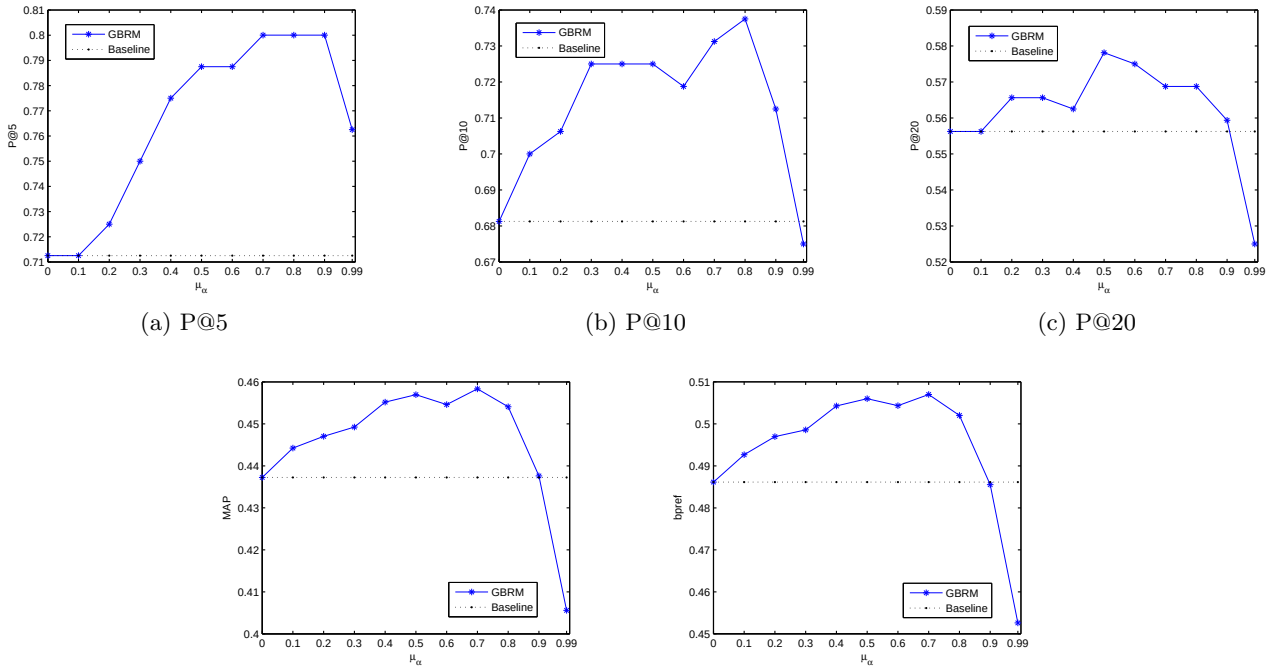
	P@5	P@10	P@20	MAP	bpref	Time
$k_{nn} = 5$	78.75	71.88	56.88	45.03	49.85	0.98s
$k_{nn} = 10$	80.00	73.13	56.88	45.84	50.70	2.11s
$k_{nn} = 20$	80.00	71.88	57.19	45.80	50.60	5.53s
$k_{nn} = 30$	78.75	71.88	56.88	45.88	50.65	9.06s
$k_{nn} = 40$	78.75	71.88	56.56	45.33	50.13	13.21s

and the dashed line denotes the *baseline*. We can see that when incorporating the global consistency in the framework ($\mu_\alpha > 0$), the performance is improved compared to the *baseline*, especially in the very top precision (P@5). With the increase of μ_α, the performance becomes better until it puts too weight on the term of global consistency ($\mu_\alpha \to 1$). As shown in Fig. 4, when the parameter μ_α is equal to 0.99, the performance of our *GBRM* method becomes worse than the initial *baseline* due to the overweighted global consistency. This observation matches the theoretical analysis of the proposed regularization framework, which leads us to believe that it is useful to incorporate global consistency over the graph. Moreover, the regularization framework is relatively robust and may achieve the best results when the parameter μ_α is set to be 0.5-0.7. The parameter μ_α used in Section 7.1 is set to be 0.7.

7.3 Effect of Graph Construction

In the previous subsections, we have shown the graph-based re-ranking model can improve the performance over the initial *baseline*. We now consider the graph construction with different settings. The objective is to investigate the effect of graph construction. We perform the experiments on the latent space graph with different settings for graph construction: 1) different dimensionality (k_d) of the latent feature, 2) different number of nearest neighbors (k_{nn}).

We utilize the joint optimization (Eq. 6) to learn a low-dimensional feature, then build the graph using the new feature as shown in Section 4. In Table 4, we show the performance for different dimensionality k_d of the feature x_i, which is used to calculate the weight w_{ij}. It occurs that the overall performances (MAP and bpref) become higher



(a) P@5 (b) P@10 (c) P@20

(d) MAP (e) bpref

Figure 4: The effect of varying the parameter μ_α.

for greater k_d. We believe this is because the higher dimensional space can better capture the similarities in the original feature space. In the table, the first row *TFIDF* is performed on the original *tf.idf* feature. In contrast, the latent feature with 50 dimensions achieves promising results than the original *tf.idf* feature, which shows the effectiveness of the latent feature.

In Table 5, we show the performance for different settings of k_{nn}, which is used to construct the adjacent matrix (graph). The last column in Table 5 denotes the average processing time for retrieving the top 1,000 experts in response to each query. The processing time increases (nearly)linearly with the increase of k_{nn}. This is because larger k_{nn} makes the graph denser, which will result in higher computational complexity in solving the Eq. 4. Comparing to the performance, we observe that it tends to degrade a little with increasing k_{nn}. In spite of this, all of those results outperform the baseline. In other experiments, we set $k_d = 100$ and $k_{nn} = 10$ which give the best results.

8. CONCLUSIONS AND FUTURE WORK

In this paper, we have described a novel and general framework which is used to model the re-ranking algorithm, by regularizing the smoothness of ranking scores over the graph, along with a regularizer on the initial ranking scores. The key to refining the results is the global consistency over the graph, which leverages the graph-based model for the query-dependent ranking problem. Empirical studies on the application of expert finding show the improvement in our proposed approach is promising. For future work, we plan to build larger query set for evaluation and pursue other applications of the proposed framework. In addition, we may extend our framework to consider the diversity of the retrieved results.

9. ACKNOWLEDGMENTS

We thank the anonymous reviewers for their helpful comments. This work is fully supported by two grants from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. CUHK4125/07E and Project No. CUHK4150/07E).

10. REFERENCES

[1] Dblp bibliography. URL:http://www.informatik.uni-trier.de/~ley/db/.

[2] Expert lists. URL:http://keg.cs.tsinghua.edu.cn/project/psn/dataset.html.

[3] A. Agarwal and S. Chakrabarti. Learning random walks to rank nodes in graphs. In *Proceedings of the 24th International Conference on Machine Learning*, pages 9–16, 2007.

[4] A. Agarwal, S. Chakrabarti, and S. Aggarwal. Learning to rank networked entities. In *Proceedings of the Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 14–23, 2006.

[5] R. Baeza-Yates, B. Ribeiro-Neto, et al. *Modern information retrieval*. Addison-Wesley Harlow, England, 1999.

[6] K. Balog, L. Azzopardi, and M. de Rijke. Formal models for expert finding in enterprise corpora. In *Proceedings of the 29th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 43–50, 2006.

[7] C. Buckley and E. M. Voorhees. Retrieval evaluation with incomplete information. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 25–32, 2004.

[8] C. J. C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. N. Hullender. Learning to rank using gradient descent. In *Proceedings of the 22nd International Conference on Machine Learning*, pages 89–96, 2005.

[9] Y. Cao, J. Liu, S. Bao, and H. Li. Research on expert search at enterprise track of trec 2005. In *Proceedings of TREC 2005*, 2005.

[10] D. Cohn and H. Chang. Learning to probabilistically identify authoritative documents. In *Proceedings of the 17th International Conference on Machine Learning*, pages 167–174, 2000.

[11] D. A. Cohn and T. Hofmann. The missing link - a probabilistic model of document content and hypertext connectivity. In *Advances in Neural Information Processing Systems*, pages 430–436, 2000.

[12] T. Davis. *Direct Methods for Sparse Linear Systems*. Society for Industrial Mathematics, 2006.

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

[14] H. Deng, I. King, and M. R. Lyu. Formal Models for Expert Finding on DBLP Bibliography Data. In *Proceedings of the 8th IEEE International Conference on Data Mining*, 2008.

[15] F. Diaz. Regularizing ad hoc retrieval scores. In *Proceedings of the 2005 ACM CIKM International Conference on Information and Knowledge Management*, pages 672–679, 2005.

[16] H. Fang and C. Zhai. Probabilistic models for expert finding. *Proceedings of the 29th European Conference on Information Retrieval (ECIR)*, 2007.

[17] T. Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22nd ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 50–57, 1999.

[18] R. Jin, H. Valizadegan, and H. Li. Ranking refinement and its application to information retrieval. In *Proceedings of the 17th International Conference on World Wide Web*, pages 397–406, 2008.

[19] J. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM)*, 46(5):604–632, 1999.

[20] O. Kurland and L. Lee. Pagerank without hyperlinks: structural re-ranking using links induced by language models. In *Proceedings of the 28nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 306–313, 2005.

[21] O. Kurland and L. Lee. Respect my authority!: Hits without hyperlinks, utilizing cluster-based language models. In *Proceedings of the 29nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 83–90, 2006.

[22] Q. Mei, D. Cai, D. Zhang, and C. Zhai. Topic modeling with network regularization. In *Proceedings of the 17th International Conference on World Wide Web*, pages 101–110, 2008.

[23] E. Minkov, W. W. Cohen, and A. Y. Ng. Contextual search and name disambiguation in email using graphs. In *Proceedings of the 29th ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 27–34, 2006.

[24] Z. Nie, Y. Zhang, J.-R. Wen, and W.-Y. Ma. Object-level ranking: bringing order to web objects. In *Proceedings of the 14th International Conference on World Wide Web*, pages 567–574, 2005.

[25] L. Page and S. Brin. The anatomy of a large-scale hypertextual web search engine. *In Proceedings of the 7th International Conference on World Wide Web*, 98, 1998.

[26] D. Petkova and W. B. Croft. Hierarchical language models for expert finding in enterprise corpora. In *18th IEEE International Conference on Tools with Artificial Intelligence*, pages 599–608, 2006.

[27] J. M. Ponte and W. B. Croft. A language modeling approach to information retrieval. In *ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 275–281, 1998.

[28] T. Qin, T.-Y. Liu, X.-D. Zhang, D.-S. Wang, W.-Y. Xiong, and H. Li. Learning to rank relational objects and its application to web search. In *Proceedings of the 17th International Conference on World Wide Web*, pages 407–416, 2008.

[29] A. Smola and R. Kondor. Kernels and regularization on graphs. *Conference on Learning Theory, COLT/KW*, 2003.

[30] C. Zhai and J. D. Lafferty. Two-stage language models for information retrieval. In *Proceedings of the 25th ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 49–56, 2002.

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

[32] B. Zhang, H. Li, Y. Liu, L. Ji, W. Xi, W. Fan, Z. Chen, and W.-Y. Ma. Improving web search results using affinity graph. In *Proceedings of the 28th ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 504–511, 2005.

[33] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf. Learning with local and global consistency. In *Advances in Neural Information Processing Systems*, 2003.

[34] D. Zhou, S. Zhu, K. Yu, X. Song, B. L. Tseng, H. Zha, and C. L. Giles. Learning multiple graphs for document recommendations. In *Proceedings of the 17th International Conference on World Wide Web*, pages 141–150, 2008.

[35] S. Zhu, K. Yu, Y. Chi, and Y. Gong. Combining content and link for classification using matrix factorization. In *Proceedings of the 30th ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 487–494, 2007.

[36] X. Zhu, Z. Ghahramani, and J. D. Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of the 20th International Conference on Machine Learning*, pages 912–919, 2003.