Knowl Inf Syst (2013) 34:109–145
DOI 10.1007/s10115-011-0472-7

REGULAR PAPER

Query-dependent cross-domain ranking in heterogeneous network

Bo Wang · Jie Tang · Wei Fan · Songcan Chen · Chenhao Tan · Zi Yang

Received: 31 December 2010 / Revised: 28 October 2011 / Accepted: 9 December 2011 /
Published online: 18 January 2012

Abstract Traditional learning-to-rank problem mainly focuses on one single type of objects. However, with the rapid growth of the Web 2.0, ranking over multiple interrelated and heterogeneous objects becomes a common situation, e.g., the heterogeneous academic network. In this scenario, one may have much training data for some type of objects (e.g. conferences) while only very few for the interested types of objects (e.g. authors). Thus, the two important questions are: (1) Given a networked data set, how could one borrow supervision from other types of objects in order to build an accurate ranking model for the interested objects with insufficient supervision? (2) If there are links between different objects, how can we exploit their relationships for improved ranking performance? In this work, we first propose a regularized framework called HCDRank to simultaneously minimize two loss functions related to these two domains. Then, we extend the approach by exploiting the link information between heterogeneous objects. We conduct a theoretical analysis to the proposed approach and derive its generalization bound to demonstrate how the two related domains could help each other in learning ranking functions. Experimental results on three different genres of data sets demonstrate the effectiveness of the proposed approaches.

Keywords Cross-domain ranking · Heterogeneous network · Latent space · Learning to rank

1 Introduction

With the emergence and rapid proliferation of web applications, ranking over heterogeneous data sources is becoming more and more important for many applications. And we often

B. Wang · S. Chen
Department of Computer Science, Nanjing University of Aeronautics and Astronautics, Nanjing, China

J. Tang (✉) · C. Tan · Z. Yang
Department of Computer Science, Tsinghua University, 100084 Beijing, China
e-mail: jietang@tsinghua.edu.cn

W. Fan
IBM T.J. Watson Research Center, New York, USA

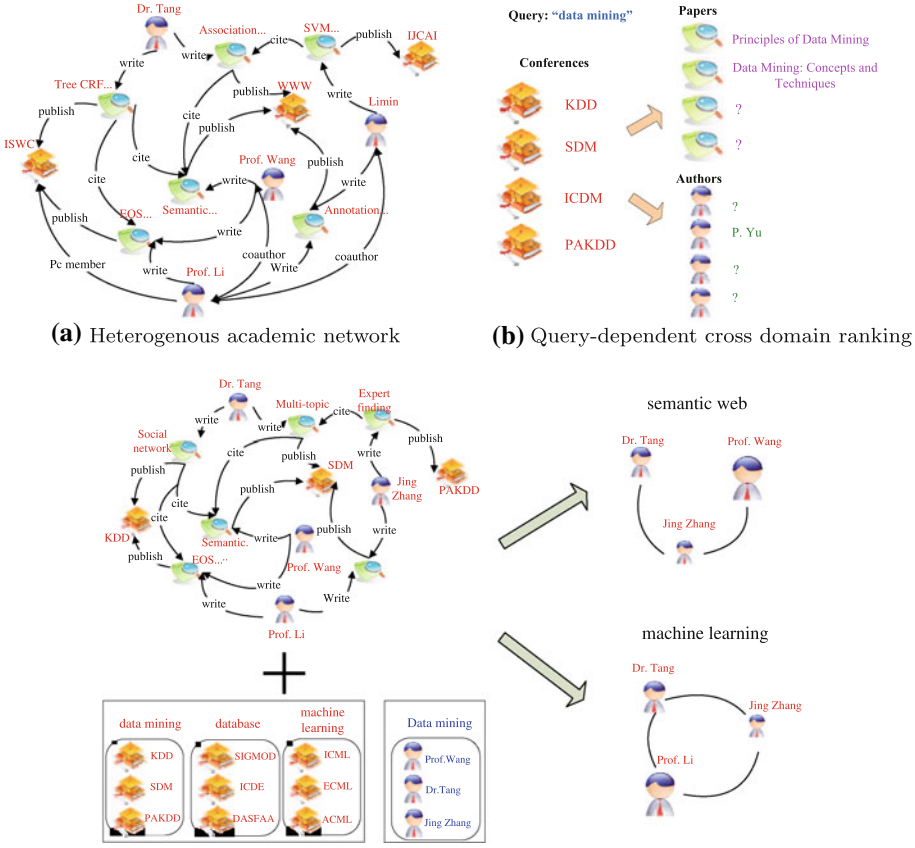
encounter the problem that we have very few data in the target domain, while we have much labeled data in the existing domains. For example, to predict the users' rating scores based on product reviews, one may have much training data (rated reviews) of existing products, but little or no training data for a new product; in social networks, we may have much training data for movie recommendation, but very few for recommending friends or web communities. Thus, one basic question is how to make use of the labeled information from existing (source) domain(s) to build an accurate ranking model for the target domain.

Although quite a few related studies have been conducted, for example, transfer learning [7,3,19,55], domain adaptation [8,9], multi-task learning [3,10], learning to rank [13,28], there are only a few theoretical studies on the heterogeneous cross-domain (HCD) ranking problem [52]. The major difference between the HCD ranking problem and learning to rank is that HCD ranking needs to consider how to borrow the preference orders from the source domain (as the supervision information) to the target domain for learning a better ranking model. The main differences between the HCD ranking problem and previously proposed transfer learning methods are as follows. First, the main focus of previous methods is on classification problems where the objective is to minimize classification error, while the proposed HCD ranking problem focuses on preference orders of multiple interested objects and there is no direct and obvious relationship between classification error and preference order-based loss functions. Second, the proposed HCDRank approach takes full advantage of the non-independent and identically distributed (i.i.d.) topological order of objects that are incorporated into the problem formulation.

1.1 Motivating application

Figure 1a shows an example of heterogenous academic search. The objective is to learn functions that can rank different objects for a given query. Figure 1b shows an example of query-dependent cross-domain ranking. For the term "query-dependent", we mean that the ranking results are specific to a query, and for different queries, the ranking results may be different. For the query "data mining", the rank levels of conferences are relatively easy to obtain (e.g., from several online resources[1]), while collecting the training data for the papers/authors would not be obvious. In this problem, we try to exploit the correlations between conferences and papers/authors for transferring knowledge from conference ranking to help learn good ranking functions for papers/authors. Figure 1c shows an example of query-dependent cross-domain ranking in heterogeneous network. In the right panel, the larger the author's icon is, the more authoritative he/she is. Besides many preference constraints over conferences under different queries, and very limited preference constraints over experts, we may also have the following relationships: coauthor relationship, author *writes* paper, paper *publishes* on a conference, and paper *cites* paper. In this problem, we try to exploit the network structure to learn a better ranking function. Intuitively, we hope that an approach can take advantage of the available supervision information (labeled conferences) and the correlation between conferences and papers/authors in the academic network to help learn the ranking functions for papers and authors. Further, if we have the information about the network structure, how can we exploit the links between heterogeneous objects for better knowledge transfer?

[1] For example, http://www.cs.ualberta.ca/~zaiane/htmldocs/ConfRanking.html.



(a) Heterogenous academic network **(b)** Query-dependent cross domain ranking

(c) Query-dependent cross domain ranking in heterogeneous network

Fig. 1 Example of query-dependent cross-domain ranking in heterogeneous network

1.2 Summaries

The challenges of query-dependent cross-domain ranking in heterogeneous network are as follows:

1. *Domain correlation.* As the types of objects in the HCD ranking problem may be different or even heterogeneous, the first challenge is how to capture the content correlation between these two different domains.
2. *Transfer ranking.* It is necessary not only to transfer the knowledge from the source domain to the heterogeneous target domain, but also to preserve the preference order with the learnt ranking model.
3. *Efficiency.* Generally, a ranking problem needs thousands (or millions) of training examples. It is important to develop methods that can scale well to large data sets.
4. *Network structure.* There are also different kinds of links between objects, so how to efficiently exploit the structure information for better knowledge transfer is important.

To address the first three challenges, we propose a unified cross-domain ranking model, named HCDRank, which simultaneously models the correlation between the source domain and the target domain, as well as learning the ranking functions. In particular, HCDRank

Table 1 List of notations

Notation	Description
G	The heterogenous network
G_S, G_T	Subgraphs for two domains
V, E	All the nodes/edges in the network
I	Index set of different types of nodes
I_S, I_T	Index sets of objects in two domains
V_S, V_T	Node sets for two domains
$\mathcal{X}_S, \mathcal{X}_T$	Instance spaces for two domains
Y_S, Y_T	Rank level sets for two domains
$\mathcal{L}_S, \mathcal{L}_T$	Labeled data in two domains
$\mathcal{A}$	Unlabeled test set in target domain
o, o'	Cardinality of the rank level set in two domains
n_S, n_T, n	# Queries of the training sets in two domains and test set
x_i	Associated attribute vector of node v_i
The following are important notations used in later Sections	
p_{ij}	The transition probability from node v_i to node v_j
f_S, f_T	Ranking functions for two domains
w_S, w_T	Weight vectors for two domains
α_1, α_2	Weight vectors for two domains in latent space
n_1, n_2	# Instance pairs in source and target domains
$\|\|W\|\|_{2,1}$	(2,1)-Norm of matrix W

uses a "latent feature space" defined over both the source and target domains to measure their correlation. Examples from both domains are mapped onto the new feature space via a projection matrix, where a common (sparse) feature space is discovered. HCDRank adopts a regularization method to simultaneously minimize two loss functions corresponding to the two domains, and supervision from the source domain is transferred to the target domain via the discovered common feature space. An efficient algorithm has been developed and a generalization bound is discussed. Regarding the fourth challenge, we propose another model called Net-HCDRank which first encodes the network structure into Markov random walk representations and then use it to augment the node-specific attribute features. Experimental results on three different types of data sets verify the effectiveness of the proposed methods, in particular when the target domain has very few labeled examples. The proposed framework is general and allows us to utilize many different algorithms to learn the ranking function.

2 Problem formulation

For ease of reading, hereafter we will denote the problem "query-dependent cross-domain ranking in heterogeneous network" by "heterogenous cross-domain ranking problem (HCD Ranking)". The HCD ranking problem can be formalized as follows. For clarity, Table 1 summarizes the notations.

Two inputs are required for our HCD ranking problem: (1) the structure of the heterogeneous network; (2) the preference constraints over a subset of the nodes.

Table 2 Three baseline methods

	RSVM	RSVMt	MTRSVM	**HCDRank**
Training data	$\mathcal{L}_T$	$\mathcal{L}_S \bigcup \mathcal{L}_T$	$\mathcal{L}_S \bigcup \mathcal{L}_T$	$\mathcal{L}_S \bigcup \mathcal{L}_T$
Test data	$\mathcal{A}$	$\mathcal{A}$	$\mathcal{A}$	$\mathcal{A}$

2.1 Heterogeneous network

In a heterogeneous network (e.g. academic social network), there are ℓ types of objects, let $I = \{1, 2, \ldots, \ell\}$ denote the type index set and V_i be the set of objects of i-th type, then the network can be represented as a graph $G = (V, E)$, where $V = \bigcup_{i \in I} V_i$ is the set of heterogeneous nodes (e.g. author, paper and conference) and $E \subset V \times V$ is the set of directed/undirected links between nodes including inter-type links and intra-type links (e.g. author publishes papers in the conferences or the coauthor relationship). For each node $v_i \in V$, there is an associated attribute vector $x_i \in \mathbb{R}^d$ which is query-dependent. For each edge $e_{ij} \in E$, there is an associated weight ω_{ij} which indicates the importance of the relationship (e.g. the total number of coauthor relationship) between v_i and v_j and $\Omega = [\omega_{ij}]$.

Let I_S denote the set of object types which can be relatively easily labeled and I_T denote the set of interested object types for which we can only collect little supervision information. From I_S, we can induce a subgraph $G_S = (V_S, E_S)$ where $V_S = \bigcup_{i \in I_S} V_i$ and $E_S \subset E$ which consists of the relationships between the nodes in V_S. Similarly, we can also define the subgraph $G_T = (V_T, E_T)$.

2.2 Query-dependent cross-domain ranking

Let $\mathcal{X}_S \in \mathbb{R}^{d_S}$ be the instance space for the source domain where d_S is the number of features, $Y_S = \{r_{S_1}, r_{S_2}, \ldots, r_{S_o}\}$ denote a set of rank levels where o is the number of rank levels. The rank levels satisfy $r_{S_1} \succ r_{S_2} \succ \ldots \succ r_{S_o}$, where $\succ$ denotes the preference relationship. The preference constraints on the subgraph G_S (the source domain) are denoted by $\mathcal{L}_S = \left\{(q_S^k, \mathbf{x}_S^k, \mathbf{y}_S^k)\right\}_{k=1}^{n_S}$, where n_S is the number of queries in the source domain. That is, for query q_S^k, $\mathbf{v}_S^k \subseteq V_S$ is the related node collection with $\mathbf{x}_S^k = \{x_{S_i}^k\}_{i=1}^{N_S^k}$ as the associated attribute vector collection and $\mathbf{y}_S^k$ as the corresponding labels where $x_{S_i}^k \in \mathcal{X}_S$, $y_{S_i}^k \in Y_S$, and N_S^k is the total number of nodes related to this query. Further, for the subgraph G_T (the target domain), let $\mathcal{X}_T \in \mathbb{R}^{d_T}$ be the instance space for the target domain where d_T is the number of features, $Y_T = \{r_{T_1}, r_{T_2}, \ldots, r_{T_{o'}}\}$ be the set of rank levels where o' is the number of rank levels. There are two parts of data in this subgraph: $\mathcal{A} = \left\{(q^k, \mathbf{x}^k)\right\}_{k=1}^{n}$ represents the unlabeled test data in which x_i^k is the attribute vector associated with node $v_i^k \in V_T$, and $\mathcal{L}_T = \left\{(q_T^k, \mathbf{x}_T^k, \mathbf{y}_T^k)\right\}_{k=1}^{n_T}$ represents the labeled data where $x_{T_i}^k \in \mathcal{X}_T$ is the attribute vector associated with node $v_{T_i}^k$, $y_{T_i}^k \in Y_T$ and n_T is the number of queries in the target domain.

2.3 Query-dependent cross-domain ranking in heterogeneous network

The HCD ranking problem can be defined as: given the network structure $G = (V, E)$, sufficient preference constraints $\mathcal{L}_S$ on the subgraph $G_S = (V_S, E_S)$, the limited number of preference constraints $\mathcal{L}_T$ and a large number of unlabeled data $\mathcal{A}$ on the subgraph $G_T = (V_T, E_T)$, the goal is to learn a ranking function f_T^* for predicting the rank levels of the unlabeled data in the subgraph G_T with the help of supervision on both subgraphs G_S and G_T.

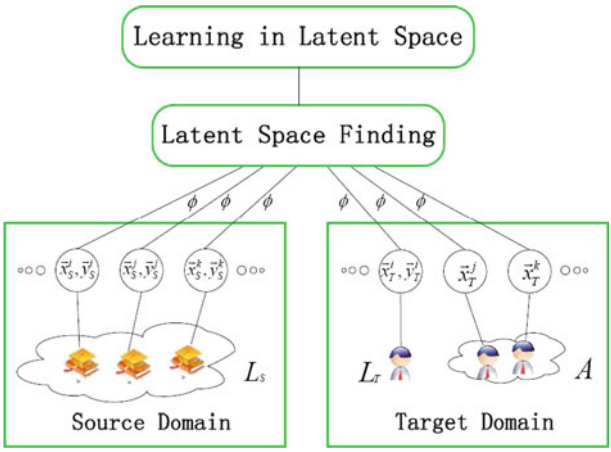


Fig. 2 Basic idea of HCD Ranking for a given query. We take the ranked conferences as the source domain and the partially ranked authors as the target domain. By using the transformation function ϕ, we can map all of them onto the latent space which can be utilized for knowledge transfer across domains

There are several key issues: (1) there exist inter-type and intra-type link structures of the two subgraphs G_S and G_T, and we need to exploit the structure information for transferring knowledge; (2) the nodes in two subgraphs may have different attribute feature distributions or even different feature spaces (e.g., different types of objects); (3) the number of rank levels in the two subgraphs can be different; (4) the number of preference constraints in two different subgraphs may be very unbalanced.

3 HCD ranking

3.1 Assumption

In HCD Ranking problem, the objects in two domains may have different feature spaces, so we first map them into a unified feature space to make them comparable. After that, the marginal distributions $p(X_S)$ and $p(X_T)$ of two domains may be still different, where X_S and X_T are constructed by the feature vectors of the nodes in V_S and V_T, respectively. So our *assumption* is that there exists a low-dimensional latent feature space determined by the transformation function ϕ, in which the marginal distributions $p(\phi(X_S))$ and $p(\phi(X_T))$ from two domains are similar.

We conduct further analysis on the heterogeneous academic data set. The conferences and experts may have different feature spaces, but both of them may focus on similar topics. That is, the research topics of an expert will have some overlaps with the target topics of the conferences. Thus, these overlapping topics will act as the low-dimensional latent space for knowledge transfer.

3.2 Basic idea

In HCD ranking, we aim at transferring preference information from an interrelated (heterogeneous) source domain to the target domain by exploiting the correlation between them. Figure 2 intuitively shows how our approach works for a given query. We take the ranked

conferences as the source domain and the partially ranked authors as the target domain. After feature extraction, all the examples with different distributions in two domains will be the inputs for the transformation function ϕ. After mapping them onto the latent space, we can find the correlation between the examples in two domain by which the knowledge can be transferred from the source domain to the target domain.

First of all, because the feature spaces of the source and target domains may be different, we use a simple way for mapping them into a unified one. Specifically, let F_c denote the set of overlapping features across domains, F_+ denote the set of features only in the target domain, F_- denote the set of features only in the source domain and $d = |F_c| + |F_+| + |F_-|$, then the instances x_S and x_T from the source and target domains in the unified feature space can be represented by $x_S = \langle F_c, F_-, \mathbf{0}_+ \rangle$ and $x_T = \langle F_c, \mathbf{0}_-, F_+ \rangle$, respectively, where $\mathbf{0}_+ \in \mathbb{R}^{|F_+|}$ and $\mathbf{0}_- \in \mathbb{R}^{|F_-|}$ are two zero vectors. Following this way, we can map the different feature spaces ($\mathbb{R}^{d_S}$ vs. $\mathbb{R}^{d_T}$) from the two domains into a unified space ($\mathbb{R}^d$).

More specifically, as the feature distributions and the objects' types may be different across domains, the first challenge is how to quantitatively measure the correlation between the different domains, which reflects what kind of information can be transferred across the domains. On the other hand, our ultimate goal is to obtain a higher ranking performance. Based on these considerations, we have two main ideas: First, we assume there is a common feature (latent) space between the two domains. Examples (e.g., x) from the two domains can be mapped onto the latent space through a transformation function $\phi(x)$. Such a common latent space provides a potential way to quantify the correlation between the two domains. Second, in the target domain, we aim to learn a ranking model that can minimize the error (loss) on the unlabeled test data while preserving the preference orders in the labeled training data. When transferring the supervision information from the source domain, we also desire to preserve its original preference orders, equivalently minimizing the loss in the source domain. Therefore, we propose a general framework (HCDRank), in which we use a latent space to bridge the two domains (i.e., the source domain and the target domain) and define two loss functions, respectively, for the two domains. We further propose an efficient algorithm to optimize the two loss functions and learn the latent space simultaneously.

3.3 The general framework: HCDRank

Given the labeled training data from the target domain $\mathcal{L}_T = \left\{(q_T^k, \mathbf{x}_T^k, \mathbf{y}_T^k)\right\}_{k=1}^{n_T}$, we aim to learn a ranking function f_T for predicting the preference relationships between instances for each query q_T^k, i.e., $f_T(x_{T_i}^k) > f_T(x_{T_j}^k) : \forall y_{T_i}^k \succ y_{T_j}^k$. For ranking, based on the learnt ranking function f_T, we can predict the rank level of a new instance. To learn the ranking function, we can consider to minimize the following loss function:

$$\begin{aligned} \min_{f_T} \mathcal{O}(f_T, \mathcal{L}_T) &= R(f_T, \mathcal{L}_T) + \eta \mathcal{E}(f_T) \\ &= \sum_{k=1}^{n_T} \sum_{y_{T_i}^k \prec y_{T_j}^k} \mathbb{I}\left[f_T\left(x_{T_i}^k\right) > f_T\left(x_{T_j}^k\right) \right] + \eta \mathcal{E}(f_T) \end{aligned} \tag{1}$$

where $\mathbb{I}[\pi]$ is the indicator function returning 1 when π is true and 0 otherwise; $R(f_T, \mathcal{L}_T)$ counts the number of mis-ranked pairs in the target domain; η is a parameter that controls the tradeoff between the empirical loss (the first term R) and the penalty (the second term $\mathcal{E}$) of the model complexity.

When transferring the supervision from the source domain, we hope to preserve the preference orders between instances from the source domain. For bridging instances from the two heterogeneous domains, we define a transformation function $\phi : \mathbb{R}^d \to \mathbb{R}^{d'}$ to map

instances from both domains to a d'-dimensional common latent space. Then, we can define a general objective function for the HCD ranking problem as follows:

$$\begin{aligned}\min_{f_S, f_T, \phi} R_\phi(f_S, \mathcal{L}_S) + C R_\phi(f_T, \mathcal{L}_T) + \lambda J_\phi(f_S, f_T) = \sum_{k=1}^{n_S} \sum_{y_{S_i}^k \prec y_{S_j}^k} \\ \times \mathbb{I}\left[f_S\left(\phi\left(x_{S_i}^k\right)\right) > f_S\left(\phi\left(x_{S_j}^k\right)\right)\right] \\ + C \sum_{k=1}^{n_T} \sum_{y_{T_i}^k \prec y_{T_j}^k} \mathbb{I}\left[f_T\left(\phi\left(x_{T_i}^k\right)\right) > f_T\left(\phi\left(x_{T_j}^k\right)\right)\right] \\ + \lambda J_\phi(f_S, f_T) \end{aligned} \tag{2}$$

where $J_\phi(f_S, f_T)$ is a penalty for the complexity of the HCD ranking model, λ is a parameter that balances the empirical losses and the penalty, and C is a parameter to control the imbalance of labeled instances across domains.

The problem now is to find the best parameters for f_S, f_T, and ϕ that minimize the objective function (Eq. 2). In the following section, we give an instantiation of the framework and present a preferred solution.

3.4 The proposed solution

In HCDRank, we do not simply want to learn the ranking function f_T, f_S for the two domains but also learn the transformation function ϕ. In addition, it is desirable to leave out features that are not important for transferring knowledge across domains and result in a sparse solution.

3.4.1 Instantiation of the HCDRank framework

For simplicity, f_T is assumed to be a linear function in the instance space: $f_T(x) = \langle w_T, x \rangle$, where w_T are parameters (feature weights) to be estimated from the training data and $\langle \cdot \rangle$ indicates the inner product. By plugging it into Eq. 1, we have

$$\mathcal{O}(f_T, \mathcal{L}_T) = \sum_{k=1}^{n_T} \sum_{y_{T_i}^k \prec y_{T_j}^k} \mathbb{I}\left[\left\langle w_T, x_{T_i}^k - x_{T_j}^k \right\rangle > 0\right] + \eta \mathcal{E}(f_T) \tag{3}$$

The loss function $R(f_T, \mathcal{L}_T)$ is not continuous, so we just use Ranking SVM hinge loss to upper bound the number of mis-ranked pairs [11].

First of all, we give a brief introduction to Ranking SVM [28]. Given the labeled data in the target domain $\mathcal{L}_T$, for each query $\mathbf{q}_T^k (k = 1, \ldots, n_T)$, given an instance pair $x_{T_i}^a, x_{T_i}^b$ from different rank levels and their corresponding labels $y_{T_i}^a, y_{T_i}^b$, Ranking SVM aims to learn a ranking function f_T which can correctly predict the preference orders between instances satisfying the following constraints:

$$y_{T_i}^a \succ y_{T_i}^b \Leftrightarrow f\left(x_{T_i}^a\right) > f\left(x_{T_i}^b\right) \tag{4}$$

Ranking SVM reduces the ranking problem into a binary classification problem by using the instance pairs as follows:

$$\left(x_{T_i}^a - x_{T_i}^b, \quad z_{T_i} = \begin{cases} +1 & y_{T_i}^a \succ y_{T_i}^b \\ -1 & y_{T_i}^a \prec y_{T_i}^b \end{cases}\right) \tag{5}$$

where $x^a_{T_i} - x^b_{T_i}$ is the new instance and z_{T_i} is the new label for the instance pair. In this way, we can get a new training data set consisting of instance pairs in the target domain $\mathcal{L}'_T = \left\{(x^a_{T_i} - x^b_{T_i}, z_{T_i})\right\}_{i=1}^{n_2}$.

Then, Ranking SVM aims to minimize the following objective function for model learning on the new training data $\mathcal{L}'_T$:

$$\min_{w_T} \sum_{i=1}^{n_2} \left[1 - z_{T_i} \left\langle w_T, x^a_{T_i} - x^b_{T_i} \right\rangle\right]_+ + \lambda \|w_T\|^2 \tag{6}$$

where the first term is called the Ranking SVM hinge loss, and the second term is the regularization term.

For the source domain, we can make the same assumption and use the parallel notations w_S and $\mathcal{L}'_S = \left\{(x^a_{S_i} - x^b_{S_i}, z_{S_i})\right\}_{i=1}^{n_1}$. Finally, we can rewrite the objective function Eq. 2 by optimizing the Ranking SVM hinge loss (the convex upper bound of the original loss) as:

$$\begin{aligned} \min_{w_S, w_T, \phi} & \sum_{i=1}^{n_1} \left[1 - z_{S_i} \left\langle w_S, \phi\left(x^a_{S_i}\right) - \phi\left(x^b_{S_i}\right)\right\rangle\right]_+ \\ & + C \sum_{i=1}^{n_2} \left[1 - z_{T_i} \left\langle w_T, \phi\left(x^a_{T_i}\right) - \phi\left(x^b_{T_i}\right)\right\rangle\right]_+ + \lambda J_\phi(w_S, w_T) \end{aligned} \tag{7}$$

Now the problem is to define the transformation function and the penalty of the model complexity.

3.4.2 Instantiation of the transformation function

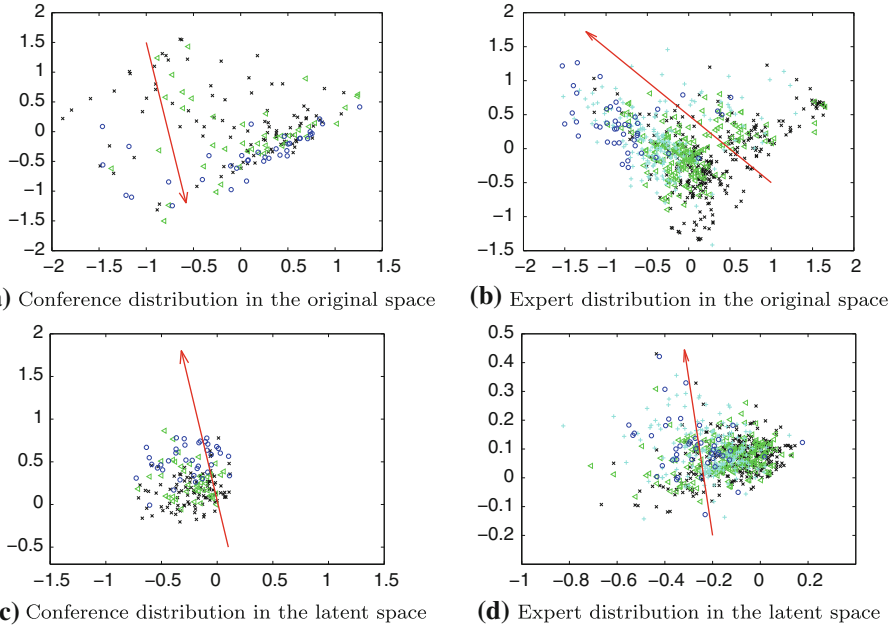
We use a $d \times d$ matrix U to describe the correlation between features, then the inner product of examples can be defined as $x_i^\top U U^\top x_j$. Such parameterization is equivalent to projecting every example x onto a latent space spanned by $\phi : x \to U^\top x$. With the transformation function, we can redefine the loss function, for example, by replacing the second term in Eq. 7 with:

$$\sum_{i=1}^{n_2} \left[1 - z_{T_i} \left\langle w_T, U^\top \left(x^a_{T_i} - x^b_{T_i}\right)\right\rangle\right]_+ \tag{8}$$

For HCD Ranking, our assumption is that there exists a low-dimensional latent space between the source domain and the target domain. For simplicity, we assume that the common features in the latent space $\phi_i(x)$, shared by the two domains, are the linear combination of the original features. That is, $\phi_i(x) = u_i^\top x$, where u_i is the orthogonal combination parameter vector, and x is the original feature vector. Further, we use U to denote the matrix with the columns the vectors $u_i (i = 1, \ldots, d)$, thus it is an orthogonal matrix. After projecting the examples from the two domains onto the latent space by matrix U, they will have similar representations.

To intuitively illustrate the semantics of the transformation function, we conduct an analysis on the academic social network data set and plot the feature distributions in different spaces for comparison in Fig. 3.

In Fig. 3a and c, black crosses, green triangles, and blue circles denote "not relevant", "marginal relevant", and "relevant" conferences, respectively. In Fig. 3b and d, black crosses, green triangles, cyan pluses, and blue circles denote "not relevant", "marginal relevant", "relevant", and "most relevant" experts, respectively. Figure 3a and b show the distributions of the conferences and experts in the original space, where we use principle component



(a) Conference distribution in the original space **(b)** Expert distribution in the original space

(c) Conference distribution in the latent space **(d)** Expert distribution in the latent space

Fig. 3 An example for illustrating the transformation function by comparison of the ranking learning in the original space and the latent space

analysis (PCA) for extracting the first two principal directions. Figure 3c and d shows their corresponding distributions in the latent space, where we use HCDRank for finding the first two principal directions. Among them, all the ranking directions (red lines with arrows) are learned by Ranking SVM. In this example, the original ranking directions for the conferences and experts differ very largely. By the transformation function, the ranking directions of the conferences and experts in the latent space have become very similar.

3.4.3 Instantiation of the penalty function

As for the penalty $J_\phi(w_S, w_T)$ of the model complexity, we define it as a regularization term, specifically, a (2,1)-norm $\|W\|_{2,1}$, for the parameters of the source and the target domains, where $W = [w_S, w_T]$ is a $d \times 2$ matrix with the first column corresponding to w_S and the second w_T. The (2, 1)-norm of W is defined as $||W||_{2,1} = \sum_{i=1}^{d} ||a^i||_2$ where a^i is the i-th row of W. The 2-norm regularizer on each row of W leads to a common feature set over the two domains and the 1-norm regularizer leads to a sparse solution. The (2,1)-norm regularizer thus offers a principled way to interpret the correlation between the two domains and also introduce useful sparsity effects. Finally, we can redefine the objective function as:

$$\begin{aligned} \min_{w_S, w_T, U} \quad & \sum_{i=1}^{n_1} \left[1 - z_{S_i} \left\langle w_S, U^\top \left(x^a_{S_i} - x^b_{S_i}\right)\right\rangle\right]_+ \\ & + C \sum_{i=1}^{n_2} \left[1 - z_{T_i} \left\langle w_T, U^\top \left(x^a_{T_i} - x^b_{T_i}\right)\right\rangle\right]_+ + \lambda \|W\|^2_{2,1} \\ s.t. \quad & U^\top U = I \end{aligned} \tag{9}$$

where $U^\top U = I$ denotes an orthogonal constraint which makes the projection matrix U unique.

Algorithm 1: HCDRank for transfer ranking

Input:
Training set: $\mathcal{L}_S \bigcup \mathcal{L}_T$; Test set: $\mathcal{A}$

Output:
Ranking function $f_T^* = \langle w_T^*, x \rangle$ and the predicted preferences over test data: $\{y_i\}_{i=1}^n$

Initialization:
$D = \frac{I_{d\times d}}{d}$

Step 1: Latent Space Finding

1: **while** not reached maximal iteration number Q **do**

2: $\alpha_1 = \texttt{argmin}\left\{ \sum_{i=1}^{n_1} \left[1 - z_{S_i} \langle \alpha, x_{S_i}^a - x_{S_i}^b \rangle\right]_+ + \lambda \langle \alpha, D^+ \alpha \rangle \right\}$

3: $\alpha_2 = \texttt{argmin}\left\{ \sum_{i=1}^{n_2} \left[1 - z_{T_i} \langle \alpha, x_{T_i}^a - x_{T_i}^b \rangle\right]_+ + \lambda \langle \alpha, D^+ \alpha \rangle \right\}$

4: $M = [\alpha_1, \alpha_2]$

5: set $D = \frac{(MM^\top)^{\frac{1}{2}}}{\texttt{Tr}(MM^\top)^{\frac{1}{2}}}$

6: **end while**

7: Apply SVD decomposition on D, $D = U \Sigma V^\top$

8: Construct U by the eigenvectors corresponding to the top two eigenvalues of D

Step 2: Learning in Latent Space

9: $w_T^* = \texttt{argmin}\left\{ \sum_{i=1}^{n_1} \left[1 - z_{S_i} \langle w, U^\top (x_{S_i}^a - x_{S_i}^b) \rangle\right]_+ + C \sum_{i=1}^{n_2} \left[1 - z_{T_i} \langle w, U^\top (x_{T_i}^a - x_{T_i}^b) \rangle\right]_+ + \lambda ||w||^2 \right\}$

10: **for** i = 1 to n **do**

11: $y_i = \langle w_T^*, U^\top x_i \rangle$

12: **end for**

3.4.4 Learning algorithm

Directly solving the objective function (including parameters w_S, w_T, U in Eq. 9) is intractable, as it is a non-convex problem. Fortunately, we can derive an equivalently convex formulation of the objective function Eq. 9 as follows: (Derivation of the equivalence is given in Appendix A).

$$\begin{aligned} \min_{M,D} & \sum_{i=1}^{n_1} \left[1 - z_{S_i} \left\langle \alpha_1, x_{S_i}^a - x_{S_i}^b \right\rangle\right]_+ + C \sum_{i=1}^{n_2} \left[1 - z_{T_i} \left\langle \alpha_2, x_{T_i}^a - x_{T_i}^b \right\rangle\right]_+ \\ & + \lambda \sum_{t=1}^{2} \left\langle \alpha_t, D^+ \alpha_t \right\rangle \\ s.t. \;\; & D \succeq 0, \quad \texttt{Tr}(D) \leq 1, \quad \texttt{range}(M) \subseteq \texttt{range}(D) \end{aligned} \tag{10}$$

where $M = [\alpha_1, \alpha_2] = UW$, $D = U\texttt{Diag}(\frac{||a^i||_2}{||W||_{2,1}})U^\top$ and the superscript "+" of D indicates the pseudoinverse of the matrix D. X is a $p \times q$ matrix, range of X is the span of columns of X which can be defined as $\texttt{range}(X) = \{x | Xz = x, \texttt{for some } z \in R^q\}$. The trace constraint of D is imposed because if D is set to ∞, the objective function will only minimize the empirical loss. The range constraint bounds the penalty term below and away from zero. The equivalence has been previously used in [3].

We can solve the equivalently convex problem with an iterative minimization algorithm, as outlined in Algorithm 1, and detailed as follows:

Step 1. We use an iterative algorithm to optimize matrix M and D. First, in lines 2-4, we keep D fixed and learn α_1 and α_2 (i.e., matrix M) from the labeled training data

in two domains, respectively. Second, in line 5, we update matrix D by the learnt matrix M. We run the above two steps iteratively until convergence or excess of the maximal iteration number. Then in lines 7 and 8, we apply SVD decomposition [51] on the learnt intermedia matrix D, i.e., $D = U\Sigma V^\top$; then, the matrix U is constructed by the eigenvectors corresponding to the top two eigenvalues of D.
The reason why we only use the two eigenvectors corresponding to the top two eigenvalues for constructing matrix U is as follows.
In algorithm 1, M is a $d * 2$ matrix, so $rank(M) \leq 2$, where $rank(M)$ is the maximum number of linearly independent column vectors of matrix M. Thus, the rank of matrix D also satisfies $rank(D) \leq 2$. After SVD decomposition on matrix D, there are at most two non-zero eigenvalues, so we just use the two eigenvectors corresponding to the top two eigenvalues for constructing the projection matrix U.

Step 2. In line 9, we learn the weight vector of the target domain from all the labeled data in the latent space. In lines 10-12, we use the learnt w_T^* to predict ranking levels of new instances from the target domain.

3.4.5 Complexity

The size of the two matrices to be optimized in HCDRank depends only on the feature number d, e.g., matrix D is $d \times d$ and W is $d \times 2$. The complexity for SVD decomposition on matrix D is $O(d^3)$. Further, for dealing with large-scale problem, PEGASUS library can be used on peta-level data [35].

Let $N = n_1 + n_2$ be the total number of instance pairs for training and s be the number of non-zero features. Using the cutting-plane algorithm [34], linear Ranking SVM training has $O\big(sN\log(N)\big)$ time complexity.

In our algorithm HCDRank, let Q be the maximal iteration number, then the training of HCDRank has $O\big((2Q+1)\cdot sN\,\mathtt{log}(N) + d^3\big)$ time complexity.

3.5 Generalization bound

In this section, we theoretically analyze our algorithm HCDRank and derive the generalization bound for it. See Appendix B for the proof.

Theorem 3.1 *Let $\mathcal{H}$ be a hypothesis space of VC-dimension k. Let $\mathcal{U}_S$ and $\mathcal{U}_T$ be unlabeled samples of size m' each, drawn from $\mathcal{D}_S$ and $\mathcal{D}_T$ respectively, and $\hat{d}_{\mathcal{H}\triangle\mathcal{H}}$ is the empirical distance between them. Let $\mathcal{L} = \mathcal{L}_S \bigcup \mathcal{L}_T$ be the labeled samples of size m generated by drawing $(1-\beta)m$ points from $\mathcal{D}_S$ and βm points from $\mathcal{D}_T$, labeling them according to f_S and f_T respectively. Let $\epsilon_S(h), \epsilon_T(h)$ be the true risks for the hypothesis h in the source and target domains respectively, $\theta = 1/(1+C)$ and $\hat{\epsilon}_\theta(h)$ be the empirical weighted risk of two domains. For each ranking function h with zero training risk, if $\hat{h} \in \mathcal{H}$ is the empirical minimizer of $\hat{\epsilon}_\theta(h)$ on $\mathcal{L}$, then with probability of at least $1-\delta$(over the choice of the samples)[8,28]*

$$\begin{aligned}\epsilon_T(\hat{h}) \leq\ & \tfrac{2}{\beta m-1}\left(k\log\left(\tfrac{8e(\beta m-1)}{k}\right)\log\left(32(\beta m-1)\right) + \log\left(\tfrac{8(\beta m-1)}{\delta}\right)\right)\\ & +2\sqrt{\tfrac{\theta^2}{\beta}+\tfrac{(1-\theta)^2}{1-\beta}}\sqrt{\tfrac{k\log(2m)-\log\delta}{2m}}\\ & +2(1-\theta)\left(\tfrac{1}{2}\hat{d}_{\mathcal{H}\triangle\mathcal{H}}(\mathcal{U}_S,\mathcal{U}_T) + 4\sqrt{\tfrac{2k\log(2m')+\log\left(\frac{4}{\delta}\right)}{m'}} + \gamma\right)\end{aligned} \tag{11}$$

where $\gamma = \min_{h\in\mathcal{H}}\epsilon_S(h) + \epsilon_T(h)$ and $\beta = \frac{n_T}{n_S+n_T}$.

The error bound is comprised of three components: the first one is the upper bound for the target risk using only the labeled data in the target domain; the second one corresponds to the difference between the true and empirical weighted risks; the last one measures the distance between target risk and weighted risk.

When we have many labeled data in the source domain and few labeled data in the target domain, θ will be small and the distance $\hat{d}_{\mathcal{H}\triangle\mathcal{H}}$ will be vital for the performance of HCDRank algorithm. In the latent space found by our algorithm, the instances from the two domains will have similar representations, then the distributions of the two domains will be similar, so $\hat{d}_{\mathcal{H}\triangle\mathcal{H}}$ will be small and the bound is tight. When the latent space is bad, the difference between these distributions is large, the bound is loose.

4 Net-HCDRank

For HCD Ranking problem, the proposed HCDRank approach transfers knowledge via the latent space by exploiting the correlations between the contents of different objects. As the rapid development of Web 2.0, the heterogenous networks are becoming more and more common where many interrelated links between different objects exist. The links are also very important for transferring ranking supervision across domains. Let us take the heterogeneous academic network as an example, there are many relationships between papers, conferences and authors, for example, author *writes* paper, and paper *publishes* on a conference and so on. The intuition for knowledge transferring by links is as follows: a paper with significant impact is very likely written by an author with significant impact, a paper with significant impact is very likely accepted by a conference with significant impact, and an author with significant impact may publish papers on a conference with significant impact. So our problem now is "while the network structure is available, how can we exploit the structural information for knowledge transfer?" In this section, we will discuss how to sufficiently exploit both content and structural information for knowledge transfer in heterogenous networks.

4.1 Basic idea

Figure 4 demonstrates the framework of the proposed Net-HCDRank algorithm. There are two phases in our algorithm: (1) feature extraction; (2) HCDRank algorithm. After extracting structural features from the heterogeneous network, the HCDRank algorithm can be applied for transferring knowledge in heterogeneous network.

While the network structure is available, there are inter-type and intra-type links across two domains (subgraphs). Sufficiently exploiting this kind of information inside the links can boost the rank performance. Meanwhile, for each node of the network $G = (V, E)$, there is an associated attribute vector $x_i \in \mathbb{R}^d$. So the first challenge is how to leverage both node-specific attribute vector and network structure to help the ranking function learning in the subgraph G_T. For phase 1, we propose a novel way to encode the network structure into Markov random walk representations based on the random walk over heterogeneous network . Then, we can use the new structural feature vector to augment the node-specific attribute feature vector for learning a better ranking model. For phase 2, we just use the HCDRank algorithm for transferring knowledge between different types of objects.

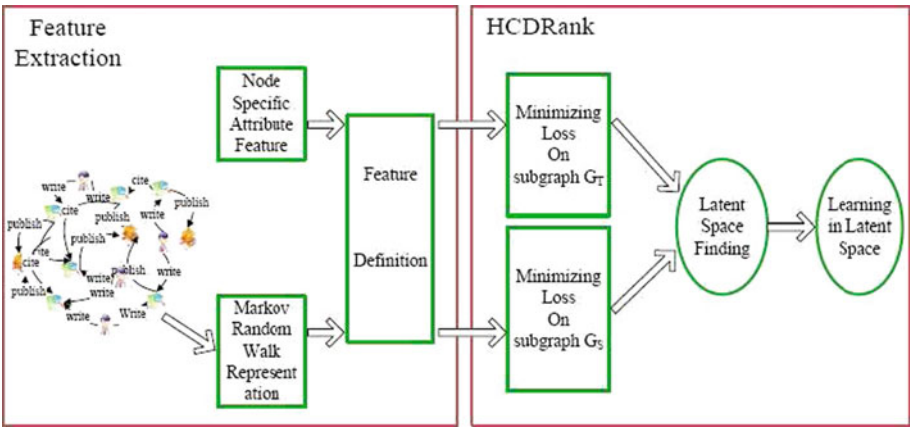


Fig. 4 Framework of the Net-HCDRank algorithm. There are two phases: feature extraction and HCDRank algorithm

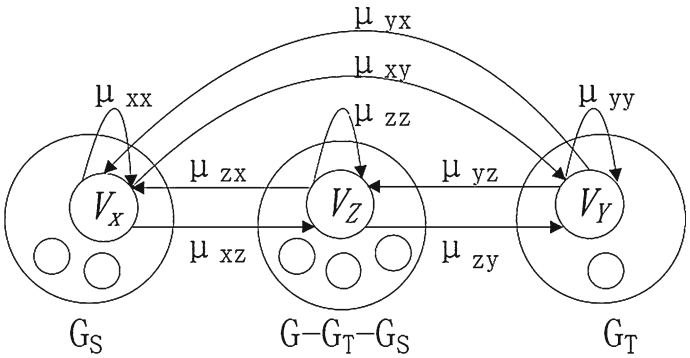


Fig. 5 Random walk over heterogeneous network

4.2 Markov random walk representation

First, we briefly define the process of random walk over the heterogeneous network. Recalling our previous definitions, the heterogeneous network $G = (V, E)$ is a directed graph with $V = \bigcup_{i \in I} V_i$ and $E \subset V \times V$ where V_i is the set of i-th type nodes.

As Figure 5 shows, there are three different collections of nodes: nodes in the subgraph G_S (the source domain), nodes in the subgraph G_T (the target domain), the other nodes in the graph $G - G_T - G_S$. There can be different types of nodes in each collection, we just use one type of nodes in each collection for illustration. For example, V_x is the node set of x-th type object in the source domain. There can be inter- and intra-links between different node collections. For example, there may exist intra-links between nodes in V_x in the source domain, and there also exist inter-links between V_x and V_y, V_z. Let the transition probability from x-th type object to y-th type object be μ_{xy}, then the transition probability from node v_i of type x to v_j of type y can be defined as

$$p_{ij} = \mu_{xy} \times \frac{\omega_{ij}}{\sum_{k \in V_y} \omega_{ik}} \quad \text{with} \quad \sum_{k=1}^{\ell} \mu_{xk} = 1 \tag{12}$$

where ℓ is the total number of different object types in the network.

Let us take the heterogeneous social network as an example for introducing the intuition of Eq. 12. After the user has found his interested expert, then with some probability he will view the expert's coauthors (intra-links), papers (inter-links), and their published conferences (inter-links). By this way, he will surf over the heterogeneous academic social network.

Regarding the μ_{xy}, a simple way for specifying its value is to calculate the ratio between the sum of edge weights from x-th type object to y-th type object and the sum of edge weights from x-th type object to all the other objects. More specifically, μ_{xy} can be specified as follows:

$$\mu_{xy} = \frac{\sum_{i \in V_x} \sum_{j \in V_y} w_{ij}}{\sum_{k=1}^{\ell} \sum_{i \in V_x} \sum_{j \in V_k} w_{ij}} \tag{13}$$

If the data set is sufficient, the values of μ_{xy} calculated in this way would be more accurate. Prior knowledge can also be utilized for this purpose.

Further, we define p_{ij} after t steps as $p_{ij}^{(t)}$, which can reflect the structure of the heterogeneous network via the probability propagation. Similar to the PageRank algorithm [16], we introduce a random jump parameter α, which allows a surfer to randomly jump to other nodes in the graph. Let $B = [p_{ij}]$ denote the one-step transition matrix, then we have

$$P = (1-\alpha)B + \alpha E, \quad E = (1, \ldots, 1)^{\top} \left(\frac{1}{|V|}, \ldots, \frac{1}{|V|} \right) \tag{14}$$

where $|V|$ is the total number of nodes in the graph. After t steps, the transition probability from v_i to v_j can be defined as $p_{ij}^{(t)} = [P^t]_{ij}$

where P^t is the t-th power of matrix P. The t-step transition probability of one node is a good representation of its network structure, so we use the transition probabilities from one node to the other nodes to form the structure feature vector associated with that node, that is, for node v_i, we use the vector $x_i' = \left(p_{i1}^t, \ldots, p_{i|V|}^t\right) \in \mathbb{R}^{|V|}$ to encode the structure information associated with this node.

Finally, we can obtain the data sets $\mathcal{L}_S = \left\{(q_S^k, \mathbf{x}_S^k, \mathbf{y}_S^k)\right\}_{k=1}^{n_S}$, $\mathcal{A} = \left\{(q^k, \mathbf{x}^k)\right\}_{k=1}^{n}$ and $\mathcal{L}_T = \left\{(q_T^k, \mathbf{x}_T^k, \mathbf{y}_T^k)\right\}_{k=1}^{n_T}$ where $x_{S_i}^k$, x_i^k and $x_{T_i}^k$ are the feature vectors associated with nodes $v_{S_i}^k$, v_i^k, and $v_{T_i}^k$, respectively, in the $\mathbb{R}^{d+|V|}$ space by concatenating x_i' into the original feature vector x_i.

The representation-based Markov random walk has been used in [48]; in their work, they use points of the same type to construct the k-nearest neighbor(k-NN) graph by a predefined distance metric. But for our problem, the network is intrinsic, the objects are heterogeneous, and the weight is defined by the relationship (or co-occurrence).

5 Experiments

Our approach is general and can be applied to various data sets. We perform the experiments on three different genres of data sets: a homogeneous data set which consists of documents from different domains; a heterogeneous data set which consists of three different types of objects; a heterogeneous task data set which consists of data from two different ranking tasks.

5.1 Evaluation measures, baseline methods

5.1.1 Evaluation measures

To quantitatively evaluate our method, we use Precision@n, MAP (mean average precision) [5] and NDCG (normalized discount cumulative gain) [30]. More evaluation measures can be found in [4].

The precision of top n results for a query is measured by precision at n which is defined as follows:

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

Average precision is defined based on the P@n to measure the accuracy of ranking results for a given query.

$$AP = \sum_{n} \frac{P@n \cdot \mathbb{I}[\texttt{document}n\texttt{is relevant}]}{\#\{\texttt{relevant documents}\}}$$

MAP is defined as the mean of all APs over test set and measures the mean precision of ranking results over all the queries. Different from MAP, NDCG gives high weights to the top ranked relevant documents. The NDCG score at position n is defined as

$$N@n = Z_n \sum_{j=1}^{n} \frac{2^{r(j)} - 1}{\log(1 + j)}$$

where $r(j)$ is the rank of j-th document, and Z_n is a normalization factor.

5.1.2 Baseline methods

We compare HCDRank with three baselines. Ranking SVM (RSVM) [28] is one of the state-of-the-art ranking algorithms for information retrieval which is designed for ranking in a single domain. For fair comparisons, we conduct two experiments with RSVM, one is to train the ranking model only by the labeled data in the target domain $\mathcal{L}_T$ and the other (called RSVMt) is to train the ranking model by the labeled data in the source domain and the target domain $\mathcal{L}_S \bigcup \mathcal{L}_T$. The third baseline is MTRSVM which is a multi-task feature learning approach using ranking SVM hinge loss. MTRSVM is adapted from [3], and aims at learning a low-dimensional representation shared by a set of multiple related tasks. MTRSVM is designed for addressing the situation where all the tasks have the same number of training data, by minimizing the total loss of all the learning tasks. However, in HCD Ranking problem, the training data of ranking tasks are very imbalanced, there are many labeled training data in the source domain while only limited labeled data in the target domain. Further, HCDRank only focuses on the ranking function learning for the target domain by learning the ranking function from all the labeled data in the latent space with cost-sensitive factor.

All the experiments are carried out on a PC running Windows XP with Dual-Core AMD Athlon 64 X2 Processor(2 GHz) and 2 G RAM. We use SVM^{light} [33] with linear kernel and default parameters to implement RSVM, RSVMt and the preference learning step of MTRSVM and HCDRank. The other part of MTRSVM and HCDRank have been implemented using Matlab 7.1, and the maximal iteration number Q of HCDRank is set to five. Also without special specification, we use the grid search to choose parameter C from $\{2^{-6}, 2^{-5}, 2^{-4}, 2^{-3}, 2^{-2}, 2^{-1}, 1, 2, 2^2, 2^3, 2^4, 2^5\}$, the results reported below are all

Table 3 Description of features in LETOR data set

Data set	Category	Description	#Features	Feature IDs
TREC	A	Low-level content features	16	{2-5,9-12,28-35}
	B	Some high-level content features	6	{15,16,19,20,23,24}
	C	Some high-level content features	7	{1,17,18,21,22,25,26}
	D	Hyperlink features	7	{6-8,14,36-38}
	E	Hybrid features	8	{13,27,39-44}
OHSUMED	F	Low-level title features	10	{1-10}
	G	Low-level abstract features	10	{11-20}
	H	High-level content features	5	{21-25}

averaged over 10 runs and we judge statistical significance using the dependent t test with $p < 0.05$.

5.2 Results on homogeneous data

5.2.1 Data set

We use LETOR 2.0 [40] as the homogeneous data set, which is a data set for evaluating various algorithms for learning to rank. LETOR 2.0 is comprised of three sub data sets: TREC2003, TREC2004, and OHSUMED, with respectively 50, 75, and 106 queries. A set of query-document pairs are collected in each of the data sets. The TREC data set is a collection from a topic distillation task which aims to find good entry points principally devoted to a given topic. The OHSUMED data set is a collection of records from medical journals. In the OHSUMED data set, there are three rank levels, i.e. relevant $\succ$ partially relevant $\succ$ non-relevant, while in the TREC data set, there are only two, i.e. relevant $\succ$ non-relevant. In LETOR, all the features are highly abstract. In TREC, there are 44 features divided into four categories. In OHSUMED, there are 25 features falling into three categories. Table 3 summarizes the features in the LETOR data set. For example, for TREC data, there are 16 low-level content features (e.g. tf and idf), 13 high-level content features (e.g. BM25 and language model for IR), 7 hyperlink features (e.g. PageRank and HITS) and 8 hybrid features (e.g. hyperlink-based relevance propagation).

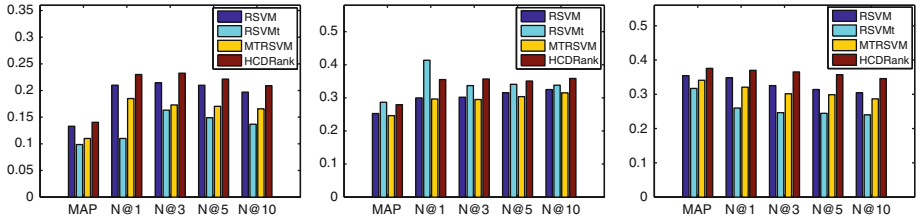
5.2.2 Feature definition

To adapt to the cross-domain ranking scenario, we make slight revision to the LETOR data set. After revision, the whole data set and three sub data sets are correspondingly referred to as LETOR_TR, TREC2003_TR, TREC2004_TR and OHSUMED_TR. Specifically, we split each data set into two domains (source domain and target domain), according to the feature types. Table 4 lists statistics of the data sets in which the 4th column shows the details of features used in each domain of every data set by feature categories A-H in Table 3. We split features in this way which can simulate some real applications. For example, the source domain of TREC2003_TR only contains feature categories A and B with queries 1-25 which correspond to features for document contents; while the target domain of TREC2003_TR consists of feature categories B, C, D, E with queries 26-50 which may correspond to features in blogs. After this splitting, intuitively the features in two domains are quite different.

Table 4 Data characteristics of LETOR_TR data set

Data Set	Domain	Query IDs	Features	#Doc	#D/Q	#Dp/Q
TREC2003_TR	SOURCE	25:{1-25}	AB	24,079	963	6,450
	TARGET	25:{26-50}	BCDE	25,092	1,004	13,761
TREC2004_TR	SOURCE	38:{1-107}	AE	37,154	978	5,969
	TARGET	37:{111-221}	BCDE	37,016	1,000	5,696
OHSUMED_TR	SOURCE	56:{1-56}	FH	8,136	145	5,726
	TARGET	50:{57-106}	GH	8,004	160	5,239

#D/Q and #Dp/Q denote the average number of documents and document pairs corresponding to a query respectively



(a) TREC2003_TR Cosine Similarity=0.01 **(b)** TREC2004_TR Cosine Similarity=0.23 **(c)** OHSUMED_TR Cosine Similarity=0.18

Fig. 6 MAP and NDCG performances for LETOR_TR

In all experiments, we use all the labeled query-document pairs from the source domain as the training data $\mathcal{L}_S$, and randomly sample 20% queries with all their labeled documents from the target domain as the training data $\mathcal{L}_T$ (that is, 5, 8, and 10 queries for TREC2003_TR, TREC2004_TR and OHSUMED_TR, respectively), while all the other data in the target domain are viewed as the unlabeled test set $\mathcal{A}$.

For ease of implementation, in this experiment, we still define each instance by a vector of 44 (TREC) or 25 (OHSUMED) dimensions. We set the feature values that are not defined in a domain as zeros. For example, in the source domain of TREC2003_TR, only features of categories A and B are set with their actual values, the values of others (B, C, D, E) are set to zeros. Similarly, in the target domain of TREC2003_TR, features in categories B, C, D, and E have their actual values and the others are zeros.

5.2.3 Results and analysis

Figure 6 and Table 5 show the results of all methods on the LETOR_TR data sets. Generally, our approach achieves higher performances and has a nice convergence property (converging after several iterations in most cases). Specifically, we have the following observations:

1. **Ranking accuracy.** HCDRank performs much better (by +5.6% and +6.1%, respectively, in terms of MAP) than the comparison methods on both TREC2003_TR and OHSUMED_TR. On TREC2004_TR, HCDRank results in a comparable performance with RSVMt.
2. **Effect of difference.** We measure the difference of the source domain and the target domain in each data set by the cosine-based similarity. We first use the mean vectors

Table 5 MAP and NDCG performances for LETOR_TR

Data Set	Baselines	MAP	N@1	N@3	N@5	N@10
TREC2003_TR	RSVM	.1330*	.2100	.2144*	.2098*	.1970*
	RSVMt	.0986*	.1100*	.1631*	.1489*	.1368*
	MTRSVM	.1100*	.1850*	.1730*	.1703*	.1656*
	HCDRank	**.1404**	**.2300**	**.2325**	**.2214**	**.2089**
TREC2004_TR	RSVM	.2526*	.3000	.3019*	.3153*	.3251*
	RSVMt	**.2866**	**.4138**	.3371	.3408	.3383
	MTRSVM	.2464*	.2966	.2949*	.3038*	.3151*
	HCDRank	.2795	.3552	**.3571**	**.3508**	**.3586**
OHSUMED_TR	RSVM	.3541*	.3483	.3255*	.3141*	.3044*
	RSVMt	.3171*	.2600*	.2465*	.2446*	.2402*
	MTRSVM	.3411*	.3208*	.3015*	.2989*	.2868*
	HCDRank	**.3758**	**.3700**	**.3654**	**.3573**	**.3459**

Best performances are in bold font, and statistically significant improvements are with asterisk(*)

of the two domains as their representatives, and then, the cosine similarity is calculated between these two mean vectors. The cosine similarities of the three sub data sets are 0.01, 0.23, and 0.18. We see that when the similarity is relatively high (0.23 on TREC2004_TR), simply combination of the training data from both domains for learning would result in a better ranking performance: RSVMt performs better than MTRSVM and RSVM. When the similarities are relatively low (0.01 on TREC2003_TR and 0.18 on OHSUMED_TR), such a brute combination will introduce a lot of noise which hurts the performance: RSVMt underperforms MTRSVM and RSVM. In both situations, our approach can balance the difference and consistently outperform the three methods.

3. **Reason for performance.** We conduct an analysis of why HCDRank is effective on LETOR_TR. An important observation is that, in the ranking problem, many features are extracted from query-document pairs, so the features already contain information from both queries and documents. Thus, a good latent space means that if the new feature representations in that space of two query-document pairs q_1-d_1 and q_2-d_2 from the two domains are similar, then the rank levels of the two documents are also similar. For example, if d_1 is relevant to q_1, then it is highly possible that d_2 is also relevant to q_2.

5.3 Results on heterogeneous data

5.3.1 Data Set

The second data set is a heterogeneous academic data set, which contains 14, 134 authors, 10, 716 papers, and 1, 434 conferences. The queries are 44 most frequent queried keywords (e.g., "data mining", "information retrieval") collected from the query log of the ArnetMiner[2] system [50]. For evaluation, we used the method of pooled relevance judgments [12] together with human judgments. Specifically, to obtain the ground truth for experts, for each query,

[2] ArnetMiner(arnetminer.org) is a free online service to search and perform data mining operations against academic publications on the Internet, using social network analysis to identify connections between researchers, conferences, and publications.

Table 6 Data characteristics of the heterogeneous academic data set

Domain	#Query	Object type	#Feature	#Object	#Object/Query
SOURCE	44	Conferences	16	1,434	33
TARGET	44	Experts	17	14,134	321

the top 30 experts from Libra[3], Rexa[4], and Arnetminer were collected, respectively, and pooled into a single list by removing the same or ambiguous ones. Then, annotators (two faculties and five graduates from CS) provided human judgments 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, what distinguished awards (Turing award, IEEE/ACM fellow and so on) he/she has been awarded. There are four rank levels (3, 2, 1, and 0), which, respectively, represent definite relevance $\succ$ relevance $\succ$ marginal relevance $\succ$ not relevance. To obtain the ground truth for conferences, the top 30 conferences from Libra and ArnetMiner are collected and three online resources[5] are mainly referenced for conference ranking. Table 6 lists the statistics of this heterogeneous academic data set.

In this experiment, we aim to answer the question: how can heterogeneous data be bridged for better ranking? We use the labeled data of one type of object (e.g., conferences) as the source domain and another type of object (e.g., authors) as the target domain. Thus, our goal is to transfer the conference ranking information for ranking authors.

5.3.2 Feature definition

We use titles of all papers published in a conference to form a conference "document" and use titles of all papers written by an author as the author's "document". Thus, we can define features for each object as listed in Table 7. For each "document", there are 10 low-level content features (e.g. L1 is term frequency(tf), L5 is inverse doc frequency(idf)) and 3 high-level content features (e.g. H1 and H2 are the original and log values of BM25 score, H3 is the value of language model for IR). S1-S3 are special features for a conference which measure the number of years held and the total citations. S4-S7 are special features for an expert, for example, the year of his first paper and the citations of all his papers. Finally, we define 16 features (L1-L10, H1-H3 and S1-S3) for conferences and 17 features for experts (L1-L10, H1-H3 and S4-S7).

We normalize the original feature vectors by query. Suppose there are $N^{(i)}$ documents $\{d_j^{(i)}\}_{j=1}^{N^{(i)}}$ for i-th query, then after normalization, the feature $x_j^{(i)}$ of document $d_j^{(i)}$ will become

$$\frac{x_j^{(i)} - \min_k \{x_k^{(i)}\}}{\max_k \{x_k^{(i)}\} - \min_k \{x_k^{(i)}\}} \quad k = 1, \ldots, N^{(i)}$$

[3] Libra(libra.msra.cn) is the academic search engine from Microsoft, which supports the search of conferences, papers and experts.

[4] Rexa(rexa.info) is a digital library and search engine covering the computer science research literature and their authors.

[5] http://www.cs.ualberta.ca/~zaiane/htmldocs/ConfRanking.html and http://www3.ntu.edu.sg/home/ASSourav/crank.htm and http://www.cs-conference-ranking.org/conferencerankings/alltopics.html

Table 7 Feature definitions for expertise search

Features	Description
L1-L10	Low-level content features, refer to [40]
H1-H3	High-level content features, refer to [40]
S1	The number of years the conference has been held
S2	The total citation of one conference during recent 5 years
S3	The total citation of one conference during recent 10 years
S4	The number of years passed since his first paper
S5	The total citation of one expert
S6	The number of papers cited more than 5 times
S7	The number of papers cited more than 10 times

Table 8 Performances of different approaches for expert finding

Approach	MAP	N@1	N@3	N@5	N@10
Libra	.5823*	.3393*	.2942*	.3054*	.3799*
Rexa	.6218*	.2560*	.2705*	.2759*	.3602*
RSVM	.8084	.6071	.5839*	.5854	.6385
RSVMt	.8096*	.5944	.6026	.5956	.6387
MTRSVM	.8059*	.5791	.5796	.5810	.6379
HCDRank	**.8195**	**.6250**	**.6257**	**.6152**	**.6615**

Best performances are in bold-font, and statistically significant improvements are with asterisk(*)

5.3.3 Results and analysis

In this experiment, we take the labeled conference data as the source domain, and the expert data as the target domain in which we use one query with its corresponding documents as the labeled data and the rest as the unlabeled test data. The results reported below are averaged over all the queries. The parameter C is empirically set to 1.

As for the baseline methods, besides RSVM,RSVMt, and MTRSVM, we also compare our approach with the results of two online academic search systems: Libra.msra.cn and Rexa.info, which are mainly based on unsupervised learning algorithm, e.g., the language model [60]. Table 8 shows the results of different approaches, the main observations are as follows:

1. **Ranking accuracy.** Among all the approaches, HCDRank outperforms the five baselines. The performances of RSVM and MTRSVM are comparable. Also all learning-to-rank methods outperform the two systems which suggest the usefulness of supervision in a specific domain for improving the ranking performance.
2. **Feature analysis.** Figure 7 shows the important features for knowledge transfer via the latent space, which can be identified by the learnt final weight vectors. We can see that the final w_T^* can exploit the data information from two domains and adjust the weights learnt from single domain to better predict preferences in the target domain. This is the major reason why the proposed method performs best. The right table in the figure lists the top 10 features vital for knowledge transfer in this academic data set by the descending order

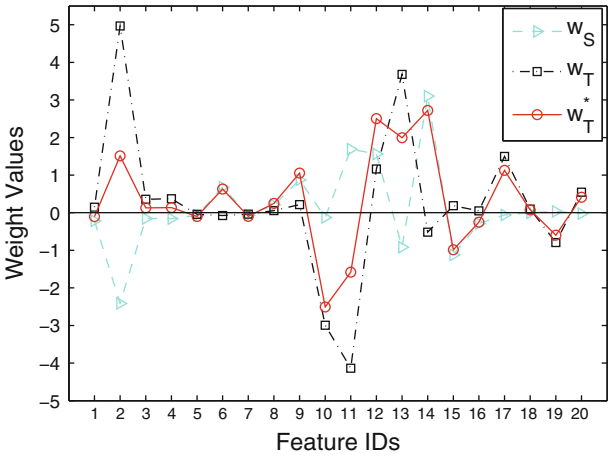


Fig. 7 Feature correlation analysis in the source and the target domains. The *red* colored weights $\mathbf{w}_T^*$ are learnt by HCDRank; the *blue* and *black* ones ($\mathbf{w}_S$ and $\mathbf{w}_T$) are learnt from the two domains separately. The table lists top 10 features learnt from the academic data set for HCD ranking

of the absolute weight values. There are L2, L6, L9, L10 in low-level content features, H1-H3 in high-level content features, and S1, S2, S4 in self-defined features.

3. **Reason for performance.** The key reason is that even in the heterogeneous network, there might be latent dependencies between the objects, some common features can still be extracted from the latent dependencies. For example, in the expertise search, authors and conferences are connected by the papers they have published. The discovered latent dependencies can be used to transfer supervised knowledge between the heterogeneous objects. Our approach can effectively discover the common latent space in the heterogeneous network thus can achieve better performance for expertise search.

5.4 Results on heterogeneous tasks

5.4.1 Data set

The third experiment is for heterogeneous tasks, where we have two different ranking tasks: expert finding and best supervisor finding. The goal of expert finding is to find experts on a given topic (query), while best supervisor finding is about finding the best supervisors in a specific domain, which is useful for junior students to find "good" supervisors in their interested fields. An expert can be a good supervisor, but not necessarily, thus the two tasks are related but different. The goal of this experiment is to evaluate whether the proposed approach can transfer knowledge to improve a different ranking task (best supervisor finding) using training data of an existing related heterogeneous ranking task (expert finding).

The evaluation data set for best supervisor finding is created by collecting the feedbacks from many researchers in related domains. The data set for best supervisor finding consists of 9 most frequent queries, for each query, we choose the top ranked 50 researchers by ArnetMiner.org and another 50 researchers who start publishing papers only in recent years ($>$2003, 91.6% of them are currently graduates or postdoctoral researchers). We sent to each of the researchers an email, in which we listed the top 50 researchers for each query, and ask for feedbacks on whether each candidate is the best supervisor ("yes") or not ("no"), or

Table 9 Data characteristics of the heterogeneous task data set

Domain	#Query	Object type	#Feature	#Object	#Object/Query
SOURCE	44	experts	21	14,134	321
TARGET	9	researchers	53	488	54

Table 10 Features for expert finding and best supervisor finding

Feature	Description
L1-L10	Low-level language model features, refer to [40]
H1-H3	High-level language model features, refer to [40]
B1	The year he/she published his/her first paper
B2	The number of papers of an expert
B3	The number of papers in recent 2 years
B4	The number of papers in recent 5 years
B5	The number of citations of all his/her papers
B6	The number of papers cited more than 5 times
B7	The numebr of papers cited more than 10 times
B8	PageRank score in academic network
SumCo1-8	The sum of coauthors' B1-B8 scores
AvgCo1-8	The average of coauthors' B1-B8 scores
SumStu1-8	The sum of his/her advisees' B1-B8 scores
AvgStu1-8	The average of his/her advisees' B1-B8 scores

"not sure". Participants can also add other best supervisors. Based on the feedbacks from the participants, we organized a list for evaluating best supervisor finding. We rated each candidate person by simply counting the number of "yes"(+1) and "no" (−1) from the received feedbacks and averaged the ratings over the number of the corresponding definite feedbacks ("yes" and "no"). In this way, we created a relatively commonly accepted best supervisor list for each query. Table 9 lists the statistics of this heterogeneous task data set.

5.4.2 Feature definition

We define 21 common features for expert finding and best supervisor finding (as shown in Table 10). Features L1-L10 and H1-H3 are scores calculated using language models, while features B1-B8 represent the expertise scores of an author from different aspects. B5-B7 are the same as S5-S7 in Table 7. More features for expert finding can be found here [63]. In addition, we define another 32 special features for best supervisor finding. SumCo1-SumCo8 represent the overall expertise of his coauthors, and we average SumCo1-SumCo8 scores over the total number of his coauthors, denoted by AvgCo1-AvgCo8. Similarly, we consider the summation and average of the expertise of only his advisees through features SumStu1-SumStu8 and AvgStu1-AvgStu8. For SumStu1-SumStu8 and AvgStu1-AvgStu8, we need identify the adviser-advisee relationship between researchers.

We employ a heuristic-based method for that. Table 11 defines four features to identify the adviser-advisee relationship. Notation n_i is the number of publications of author i, and n_{co} is the number of cooperation publications, t_i is the year of author i's first publication, and t_{co} is the first year of coauthors' cooperations. Notation N is a constant that describes

Table 11 Features for relationship identification

Feature	Description	Formula
f_1	Coauthor paper ratio	$\frac{n_{co}}{n_i} - \frac{n_{co}}{n_j}$
f_2	Absolute paper difference	$g\left(\frac{n_i - n_j}{N}\right)$
f_3	Year of first paper	$g\left(\frac{t_j - t_i}{T}\right)$
f_4	Time interval until cooperation	$g\left(t_{co} - t_i\right) - g\left(t_{co} - t_j\right)$

Table 12 Results of best supervisor finding

Approach	P@5	P@10	P@15	MAP	N@5	N@10
RSVM	.7714	**.8429**	.8285	.7756	.5545	.5947
RSVMt	.8000	.8286	.8476	.7837	.5923	.5999
MTRSVM	.8000	.8286	.8476	.7875	.6140	.6075
Language model	.6250*	.6875*	.6500*	.6726*	.3343*	.3809*
HCDRank	**.8285**	.7857	**.8571**	**.7971**	**.6189**	**.6112**

Best performances are in bold-font, and statistically significant improvements are with asterisk(*)

the average difference of number of publications between an ordinary teacher and a student, and T is the time interval between their first publications. We take $N = 10$ and $T = 10$ in our experiments. $g(x)$ is an identity function if $-1 < x < 1$ and a sign function if $x \leq -1$ or $x \geq 1$. For any two researchers i and j, we calculate a score $s_{ij} = \sum_k \lambda_k f_k(i, j)$, where weight $\{\lambda\}$ of the features is predefined. Finally, if $s_{ij} > r$, we say author i is the advisor of author j; if $s_{ij} < -r$, we say author i is advised by author j, where r is a predefined threshold, and usually takes 2.5∼3.5. Experiments show that the accuracy of relationship identification with this method is 67.0%. Interested readers can refer to [58] for our demo.

5.4.3 Results and analysis

In this experiment, for the source domain data, we use all the labeled data from the expert finding task, and for the target domain data, we use two sampled queries with their corresponding documents from the best supervisor finding task as the labeled data, and the rest as the unlabeled test data. Table 12 shows the performance of best supervisor finding. We see that the proposed method performs better than the baseline methods of using RSVM, RSVMt, MTRSVM and the language model-based method [60]. Also we can see that all supervised learning-to-rank methods can achieve higher ranking accuracies than the unsupervised ranking method (language model).

Table 13 show the top 5 best supervisors/experts for two example queries. From that, we can see the traditional expert finding algorithm is inappropriate for the best supervisor finding task.

5.5 Results on heterogenous data with network structure

5.5.1 Data set

There is no existing publicly available data set in our problem setting, so we only perform our experiment on the heterogeneous academic dataset in the second experiment with minor

Table 13 Example lists of expert finding verse best supervisor finding

Best supervisor finding		**Expert finding**	
Machine learning	SVM	Machine learning	SVM
Geoffrey E. Hinton	Bernhard Scholkopf	Pat Langley	Bernhard Scholkopf
Sanjay Jain	Vladimir Vapnik	Ivan Bratko	Vladimir Vapnik
Michael I. Jordan	John Shawe-Taylor	Thomas G. Dietterich	Olvi L. Mangasarian
Tom M. Mitchell	Alex J. Smola	Carl H. Smith	Chih-Jen Lin
Avrim Blum	Thomas Hofmann	Jaime G. Carbonell	Thorsten Joachims

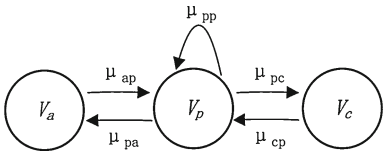


Fig. 8 Random walk over heterogeneous network

modification. Because there is no link information in the original dataset, we utilize Arnet-Miner for generating the links, but there are some authors who cannot be matched perfectly, so we remove them. Finally, the academic data set contains 14,023 authors, 10,212 papers, and 1,382 conferences. In this experiment, we aim to transfer preference constraints from conference subgraph to author subgraph via both the content correlation and the structure of the whole network.

5.5.2 Feature definition

There are two different parts of features: (1) for node-specific attribute feature, we use the same definition as Table 7; (2) for network structure feature, we will define them later in this subsection. We have three different types of objects: paper(p), author(a) and conference(c). In this heterogeneous network, for nodes, we use all of the three different types of objects; for edges, we use the paper citation relationship, coauthor relationship, the author *writes* paper relationship, the paper *publishes* on the conference relationship; for weights, if there is an edge between two nodes, then the weight is 1, otherwise 0. Further, the set I_S only contains the conference object type, and I_T only contains the author object type.

We use Fig. 8 to demonstrate the inter-type and intra-type link structure of the heterogeneous academic network. Now the problem is how to calculate the transition probabilities between nodes. Specifically, let V_a, V_p, V_c denote the set of authors, papers and conferences. Because there are different kinds of links between heterogeneous objects, we define different types of transition probabilities. Let μ_{pp} denote the transition probability for paper citations, μ_{pa} and μ_{ap} for paper-author relationship, μ_{pc} and μ_{cp} for paper-conference relationship. Clearly, we need $\mu_{pp} + \mu_{pa} + \mu_{pc} = 1$ and we define $\mu_{ap} = 1$ and $\mu_{cp} = 1$. Then, the transition probability from node v_i to node v_j can be defined as follows (more details can be found in [49]):

$$p_{ij} = \mu_{xy} \times \frac{1}{Out\_Degree(v_i \to V_y)} \tag{15}$$

where $x, y \in \{a, p, c\}$ are the types of node v_i and v_j; $Out\_Degree(v_i \to V_y)$ is the total number of directed edges from v_i to all the nodes in V_y.

Table 14 Feature definitions for heterogenous data with network structure

Features	Description
L1-L10	Low-level content features, refer to [40]
H1-H3	High-level content features, refer to [40]
S1-S7	Special features for conferences and experts, refer to Table 7
M1-MK	Structural features by Markov random walk representation

We use the Markov random walk representation at step t in Sect. 4 to define the network structure features. There are 25,617 different nodes in total for this heterogeneous academic network data, so we did not directly use all the structure features. After we obtain the transition matrix at step t, we can easily calculate the authority scores for each node by simply summing up the corresponding column. Then, we just use the top K nodes with highest authority scores as the representatives of the whole network. Specifically, for node v_i, we just use the transition probabilities from it to the top K nodes at step t to form the node-specific structure feature vector.

To summarize, Table 14 lists all the features for the this experiment. Among them, the first 20 features (L1-L10, H1-H3, S1-S7) are the same as Table 7, M1-MK are K structural features by Markov random walk representation.

5.5.3 Results and analysis

In this experiment, we use the same setting as Sect. 5.3. Empirically, we set the cost-sensitive parameter $C = 1$, the step parameter $t = 4, K = 100$. By analyzing the log data of ArnetMiner, we set the transition probabilities $\mu_{pp} = 0.7, \mu_{pa} = 0.29$, and $\mu_{pc} = 0.01$. Specifically, we first split the log data into different sessions of users, and then in each session, we can further analyze the actions of the user: viewing conferences, viewing papers, or viewing experts. Finally, we can count the co-occurrence of any two actions and get the transition probabilities. x

As for the baselines, besides RSVM, RSVMt, and MTRSVM, we also compare the performance of our approach with Net-RSVM, Net-RSVMt, and Net-MTRSVM which are the corresponding structural versions of those baselines by exploiting network structural information. Table 15 shows the results of different approaches, and the main observations are as follows:

1. **Ranking accuracy.** Note that the results for RSVM, RSVMt, MTRSVM, and HCD-Rank are a little different from Table 8, that is, because we remove some authors who can not be matched perfectly. Among all the approaches, Net-HCDRank outperforms the other baselines in most measures which verify the effectiveness of our approach. Generally speaking, all the structural versions of the baselines outperform the original ones in most measures which implies that exploiting the network structural information can significantly boost the performances.
2. **Feature analysis.** Figure 9 shows the final weight vectors learnt in this data set. We can see that the final w_T^* can exploit the data information from two domains and adjust the weights learnt from single domain data to better predict preferences in the target domain. This is the major reason why the proposed method performs best. The right table in the figure lists the top 15 features vital for knowledge transfer in this academic data set by the descending order of the absolute weight values.

Table 15 Performances of different approaches for expert finding

Approach	MAP	N@1	N@3	N@5	N@10
RSVM	.8457	**.6811**	.6228	.6083	.6695
RSVMt	.8431*	.5995*	.6001	.5885	.6501*
MTRSVM	.8439	.6709	.6224	.6071	.6651
HCDRank	**.8533**	.6480	.6323	.6188	.6812
Net-RSVM	.8422	.6760	*.6400*	*.6199*	*.6701*
Net-RSVMt	.8374*	*.5995**	*.6122*	*.5989*	.6446*
Net-MTRSVM	.8386	.6454	*.6235*	.6045	.6613
Net-HCDRank	.8508	*.6760*	**.6428**	**.6318**	**.6816**

Best performances are in bold-font, the improved performances by exploiting the network information are in italic font, and statistically significant improvements are with asterisk(*)

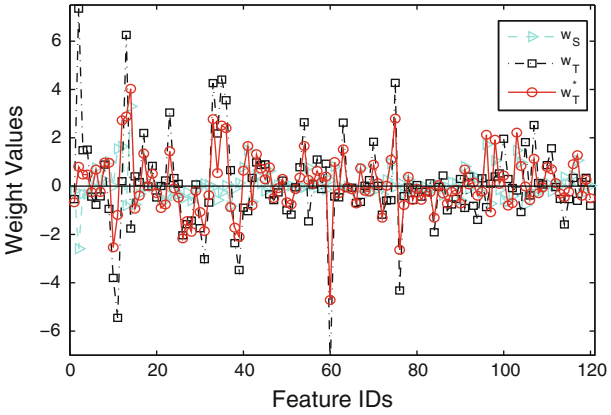
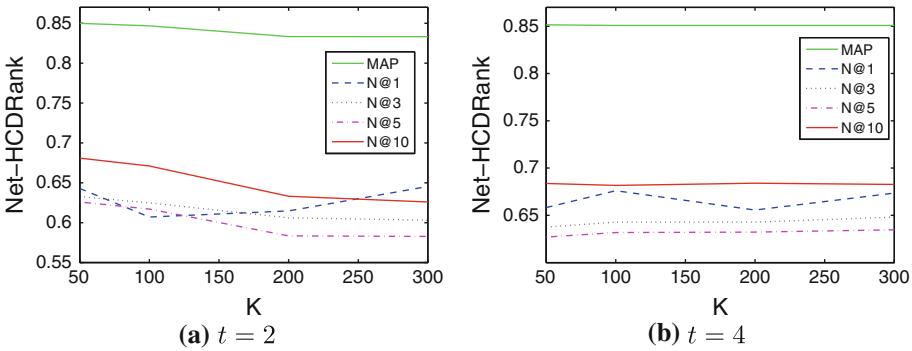


Fig. 9 Feature correlation analysis in the source and the target domains. The *red* colored weights $\mathbf{w}_T^*$ are learnt by Net-HCDRank; the *blue* and *black* ones ($\mathbf{w}_S$ and $\mathbf{w}_T$) are learnt from the two domains separately. The table lists top 15 features learnt from the academic data set for Net-HCDRank

Among them, there are one low-level content feature (L10), two high-level content features (H2, H3), one special feature for conference (S1). All the other 11 features are extracted from network structure, that is, the network structure information is very important for transferring knowledge across domains.

3. **Parameter sensitivity analysis.** Figure 10 shows how the measures change while the parameters K and t vary. From this figure, we can mainly conclude that: (1) When $t = 2$, as K varies, all the measures (MAP, N@1, N@3, N@5, N@10) are relatively not stable, and the performance drops relatively sharp, which indicates that the score after 2 steps may include more noise. (2) When $t = 4$, as K varies, all the measures (MAP, N@1, N@3, N@5, N@10) are very stable and very close. That means, the random walk results at step 4 are more reasonable to describe the network structure.
4. **Reason for performance.** The academic data have intrinsic link structure which is very useful for ranking objects. We try to exploit this information by Markov random walk representation. The supervision can be propagated from the source subgraph to the target subgraph via the links within the whole network. That is the main reason why Net-HCDRank can outperform HCDRank significantly.



(a) $t = 2$ **(b)** $t = 4$

Fig. 10 How performances vary as K increases

6 Related work

6.1 Learning to rank

Considerable works have been conducted for supervised learning to rank [32,6], which can be divided into three categories: pointwise approach, pairwise approach [46], and listwise approach. In pointwise approaches, the ranking problem aims at predicting the rank level of an object. In pairwise approaches, the ranking problem can be reduced to binary classification by comparing the rank levels of instance pairs. Ranking SVM [28], RankBoost, and RankNet [13] are three state-of-the-art algorithms in this category. In listwise approaches, the ranking problem is formulated to directly optimize some listwise performance measures of information retrieval [56,59,61].

Regarding the unavailability of a large amount of training data, there are also some works for ranking by semi-supervised learning and transductive learning. For example, Duh and Kirchhoff propose a framework for ranking in the transductive setting. They try to extract query-specific features in order to learn a query-specific ranking function [20]. Amini et al. [2] propose a semi-supervised rankboost algorithm. Hoi and Jin propose a semi-supervised ensemble ranking with a SVM-like formulation [29].

Some work takes the relations between objects to be ranked into consideration. Agarwal et al. [1] propose a framework for ranking networked entities based on a maximum entropy flow algorithm with weight function for different types of edges. In their work, they combine pairwise preference constraints into a Markov random walk process and use the total amount of in-flow to rank the entities. Our work is different from theirs in three folds: (1) we have query information; (2) we have attribute vector for each node; (3) our work is in a transfer learning setting. Qin et al. [44] propose a problem referred to as "learning to rank relational objects" while the ranking function takes the relationships between the objects to be ranked into consideration, they also formulate it into an optimization problem based on Ranking SVM for solving it. They need to predefine an outer ranking function based on relations, so it cannot be used to other tasks different from the two tasks defined in that paper, that is also the reason why we do not use it as a baseline method. The major difference from our work is that our data objects are in a heterogeneous network under transfer learning setting.

Also there are quite a few work done in cross-domain ranking problem [24,25]. Chen and Lu et al. [15] propose a tree-based ranking adaptation algorithm, aiming to make use of the training data from an existing domain. Wang et al. [52] propose a novel problem called heter-

ogeneous cross-domain ranking problem and also propose a unified regularized framework for solving it. Chapelle et al. [14] propose a novel multi-task learning algorithm with boosted decision trees for jointly learning different web search ranking tasks. Another noting thing is that the second track of Yahoo! Learning to Ranking Challenge[6] is for transfer ranking. In the second track, there are two data sets for this challenge, each corresponding to a different country. Both data sets are related, but also different to some extent. The larger set serves as the source domain and the smaller one as the target domain.

The biggest difference from our work is that our main focus is on the knowledge transfer across heterogeneous objects (eg., conferences vs. experts).

6.2 Transfer learning

Another related work is transfer learning, which aims to transfer knowledge from a source domain to a related target domain. Two fundamental issues in transfer learning are "what to transfer" and "when to transfer" [43]. Many approaches have been proposed by reweighting instances in source domain for the use in target domain [17]. Gao et al. [22] propose a locally weighted ensemble framework which can utilize different models for transferring labeled information from multiple training domains.

Also many works have been done based on new feature representation [31,36]. For example, Argyriou and Evgeniou propose a method to learn a shared low-dimensional representation for multiple related tasks [3]. The algorithm can learn the features and the task functions simultaneously in a convex optimization formulation which can be solved very efficiently. Blitzer et al. [9] propose a structural correspondence learning (SCL) approach to induce correspondences among features across two domains. Raina et al. [45] propose to use large amount of unlabeled data in source domain to improve the performance on target domain in which there are few labeled data. Xie et al. [55] propose a framework called LatentMap for dealing with the transfer learning problem where the spaces of two domains are at most overlapping, the marginal and conditional distributions are different, and the dimensionality can be extremely high. Gupta et al. [26] propose a novel nonnegative shared subspace learning framework for improving the retrieval performance by leveraging a second data source.

Some works about knowledge transfer across heterogenous feature spaces have been done [38]. For example, Yang et al. [57] propose an approach called annotation-based probabilistic latent semantic analysis(aPLSA) for boosting unsupervised learning with the help of another heterogeneous data collected from the social Web.

Shi et al. [47] propose a novel algorithm called HeMap for borrowing supervised information from data set with different feature spaces, distributions, and output spaces via spectral embedding.

Recently, there are some works done about knowledge transfer in heterogeneous network [23,27].

Dai et al. [18] propose a general framework called EigenTransfer for tackling many existing transfer learning problems. They learn the spectra of a graph consisting of features, instances, and class labels by normalized cut, then based on the spectral representation, they learn a classifier from all the labeled data in the target domain.

They transfer the knowledge across domains via constructing a task graph. That is different from ours in two folds: (1) we are dealing with heterogeneous network which is in an inherent graph setting; (2) we need to exploit the link structure between objects to boost the performance.

[6] http://learningtorankchallenge.yahoo.com.

There are also many other works which transfer information by shared parameters [10,21], relational knowledge [41], or kernel method [62]. Transfer learning techniques are widely used in classification, regression, clustering, dimensionality reduction [53], collaborative filtering [37], sentiment classification [42], and information extraction [54].

7 Conclusion and future work

We formally define the problem of heterogeneous cross-domain (HCD) ranking and address four challenges: (1) how to formalize the problem in a *unified* and *principled* framework even when the types of objects across domains are different; (2) how to transfer the knowledge of heterogeneous objects across domains; (3) how to preserve the preference relationships between instances across heterogeneous data sources; (4) how to efficiently exploit the structure information for better knowledge transfer. To address these, we propose a general regularized framework to discover a latent space for two domains and minimize two weighted ranking functions simultaneously in the latent space. We solve this problem by optimizing the convex upper bound of the non-continuous loss function and derive its generalization bound. Further, we try to exploit the network link structure for better knowledge transferring between objects with different types by Markov random walk representation. Experimental results on three different genres of data sets demonstrate the effectiveness of the proposed methods.

There are several directions for future work. It would be interesting to develop new algorithms under the framework and to reduce the computing complexity for online application. Another issue is, if there are no supervision in the target domain, so how to transfer ranking information from the source domain by the link structure of the network.

Another potential issue is to apply the proposed approach to other applications (e.g., recommendation, rating, and link prediction) to further validate its effectiveness.

Acknowledgments Songcan Chen and Bo Wang are partially supported by the Natural Science Foundation of China (No. 60973097) and Chinese National Key Foundation Research (No. 61035003). Jie Tang is supported by the Natural Science Foundation of China (No. 61073073, No. 60973102) and Chinese National Key Foundation Research (No. 60933013, No. 61035004).

Appendix A: Derivation of the equivalent convex formulation

For completeness, we give a brief proof on the equivalence between Eqs. 9 and 10. We follow the same structure as the proof of equation equivalence in [3]. For easy explanation, we denote the objective functions in Eqs. 9 and 10 as $\mathcal{E}(W, U)$ and $\mathcal{R}(M, D)$, respectively.

Theorem A.1 *Problem of* $\min\{\mathcal{E}(W,U) : U^\top U = I\}$ *is equivalent to the problem* $\min\{\mathcal{R}(M, D) : D \succeq 0, \mathtt{Tr}(D) \leq 1, \mathtt{range}(M) \subseteq \mathtt{range}(D)\}$.

Proof The correspondence between the two problems is $M = UW$, $D = U\mathtt{Diag}\big(\frac{||a^i||_2}{||W||_{2,1}}\big)U^\top$. Let a^i be the i-th row of W, $\|a^i\|_2 = \|M^\top u_i\|_2$. So

$$\sum_{t=1}^{2} \langle \alpha_t, D^+\alpha_t\rangle = \mathtt{Tr}(M^\top D^+ M) = \|W\|_{2,1}\ \mathtt{Tr}\big(M^\top U\mathtt{Diag}(\|M^\top u_i\|_2)^+ U^\top M\big)$$
$$= \|W\|_{2,1}\ \mathtt{Tr}\Big(\sum_{i=1}^{d}(\|M^\top u_i\|_2)^+ M^\top u_i u_i^\top M\Big) = \|W\|_{2,1}\sum_{i=1}^{d}\|M^\top u_i\|_2 = \|W\|_{2,1}^2$$

Therefore, $\min_{M,D}\ \mathcal{R}(M, D) \leq \min_{W,U}\ \mathcal{E}(W, U)$.

On the other side, let $D = U\mathtt{Diag}(\lambda_i)U^\top$, then

$$\sum_{t=1}^{2} \langle \alpha_t, D^+\alpha_t \rangle = \mathtt{Tr}\left(M^\top U\mathtt{Diag}(\lambda_i^+)U^\top M\right) = \mathtt{Tr}\left(\mathtt{Diag}(\lambda_i^+)WW^\top\right) \geq \|W\|_{2,1}^2$$

Hence, $\min_{M,D}\ \mathcal{R}(M, D) \geq \min_{W,U}\ \mathcal{E}(W, U)$. So they are equivalent. □

Appendix B: Proof for the generalization bound of HCDRank

Proof First of all, some notations will be introduced. A domain is defined by two terms: the distribution $\mathcal{D}$ on instance space $\mathcal{X}$, and a ranking function $f : \mathcal{X} \to \mathbb{R}$. Then, the source and target domains are denoted by $\langle \mathcal{D}_S, f_S \rangle$ and $\langle \mathcal{D}_T, f_T \rangle$, respectively.

Following [8], suppose f^* is the ideal ranking function for the target domain, and $h : \mathcal{X}_T \to \mathbb{R}$ is a hypothesis for instance space $\mathcal{X}_T$, then according the target domain distribution $\mathcal{D}_T$, the probability of h disagreeing with f^* can be defined as follows which is also the risk of this hypothesis:

$$\epsilon_T(h, f^*) = \mathbf{E}_{x_1,x_2\sim\mathcal{D}_T}\left[c_{pref}\left(x_1, x_2, h(x_1), h(x_2)\right)\right] \tag{16}$$

$$where\quad c_{pref}\left(x_1, x_2, h(x_1), h(x_2)\right) = \begin{cases} 1 & y_1 \succ y_2 \wedge h(x_1) \leq h(x_2) \\ 1 & y_1 \prec y_2 \wedge h(x_1) \geq h(x_2) \\ 0 & \text{otherwise} \end{cases}$$

in which y_1, y_2 are the corresponding rank levels of x_1, x_2.

This risk can be abbreviated to $\epsilon_T(h)$ and the corresponding empirical risk is denoted as $\hat{\epsilon}_T(h)$. The parallel notations $\epsilon_S(h, f^*)$, $\epsilon_S(h)$, $\hat{\epsilon}_S(h)$ are used for the source domain. These can be used to calculate the distance between the two domains $\mathcal{D}_S$ and $\mathcal{D}_T$ by a hypothesis class specific measure. Let $\mathcal{H}$ be a hypothesis class for instance space $\mathcal{X}$, $\mathcal{A}_\mathcal{H}$ be the subset of $\mathcal{H}$ consisting of each hypothesis $h \in \mathcal{H}$, $\left\{x_1, x_2 : x_1, x_2 \in \mathcal{X}, y_1, y_2 \in Y,\ \mathbb{I}[y_1 \succ y_2] * \mathbb{I}[h(x_1) > h(x_2)] + \mathbb{I}[y_1 \prec y_2] * \mathbb{I}[h(x_1) < h(x_2)] = 1\right\} \in \mathcal{A}_\mathcal{H}$ where y_1, y_2 are the rank levels of x_1, x_2.

Then the distance between the source and target domains is defined as

$$d_\mathcal{H}(\mathcal{D}_S, \mathcal{D}_T) = 2 \sup_{A\in\mathcal{A}_\mathcal{H}} \left|\Pr_{\mathcal{D}_S}[A] - \Pr_{\mathcal{D}_T}[A]\right| \tag{17}$$

According to [8], when $\mathcal{H}$ has finite VC-dimension k, $d_\mathcal{H}$ can be computed from finite unlabeled samples of the two domains. Let $\mathcal{H}$ be the hypothesis space, we can define the symmetric difference hypothesis space $\mathcal{H}\triangleright\!\triangleleft\mathcal{H}$ as

$$\mathcal{H}\triangleright\!\triangleleft\mathcal{H} = \left\{c(x_1, x_2, h, h') = 1 : h, h' \in \mathcal{H}\right\} \tag{18}$$

$$where\quad c(x_1, x_2, h, h') = \begin{cases} 1 & h(x_1) > h(x_2) \wedge h'(x_1) \leq h'(x_2) \\ 1 & h(x_1) < h(x_2) \wedge h'(x_1) \geq h'(x_2) \\ 0 & \text{otherwise} \end{cases} \tag{19}$$

For a pair of hypotheses in $\mathcal{H}$, if they disagree with each other on a point, then that point will be labeled as positive by each hypothesis $g \in \mathcal{H}\triangleright\!\triangleleft\mathcal{H}$. Similarly, $\mathcal{A}_{\mathcal{H}\triangleright\!\triangleleft\mathcal{H}}$ can be defined as the set of A such that $A = \left\{x_1, x_2 : x_1, x_2 \in \mathcal{X}, \mathbb{I}[h(x_1) > h(x_2)] \neq \mathbb{I}[h'(x_1) > h'(x_2)]\right\}$ for

some $h, h' \in \mathcal{H}$. Then, the distance $d_{\mathcal{H}\triangle\mathcal{H}}$ can be proven satisfying the following inequality for any hypotheses $h, h' \in \mathcal{H}$:

$$\left|\epsilon_S(h, h') - \epsilon_T(h, h')\right| \leq \frac{1}{2} d_{\mathcal{H}\triangle\mathcal{H}}(\mathcal{D}_S, \mathcal{D}_T) \tag{20}$$

Further, for the combined source and target task, the ideal hypothesis which can minimize the combined risk can be defined as

$$h^* = \underset{h\in\mathcal{H}}{\texttt{argmin}}\ \epsilon_S(h) + \epsilon_T(h) \tag{21}$$

This combined risk can be denoted as $\lambda = \epsilon_S(h^*) + \epsilon_T(h^*)$.

Regarding our problem, an equivalent formulation for Eq. 9 is as follows:

$$\begin{aligned} \min_{W,U} & \sum_{i=1}^{n_1} \left[1 - z_{S_i} \left\langle w_S, U^\top \left(x^a_{S_i} - x^b_{S_i}\right)\right\rangle\right]_+ \\ & + C \sum_{i=1}^{n_2} \left[1 - z_{T_i} \left\langle w_T, U^\top \left(x^a_{T_i} - x^b_{T_i}\right)\right\rangle\right]_+ \\ s.t.\ & ||W||_{2,1} \leq \kappa, \quad U^\top U = I \end{aligned} \tag{22}$$

where $\kappa \geq 0$ and there is a one-to-one correspondence between λ and κ [39].

In Eq. 22, the objective function is $\hat{\epsilon}_S(h) + C\hat{\epsilon}_T(h)$ with parameter $C \in [0, \infty)$. It is easy to prove that C is equivalent to the ratio $(1-\theta)/\theta$ with $\theta \in [0, 1]$, that is, $\theta = 1/(1+C)$. Thus, by replacing C with $(1-\theta)/\theta$ and multiplying both sides of the equation by θ, we can obtain the following equivalent objective function which is a convex combination of empirical source and target risk:

$$\hat{\epsilon}_\theta(h) = \theta\hat{\epsilon}_T(h) + (1-\theta)\hat{\epsilon}_S(h) \tag{23}$$

where $\theta = \frac{1}{1+C}$, $\hat{\epsilon}_\theta(h)$ and $\epsilon_\theta(h)$ are the empirical/true weighted risks respectively. Hereafter, we will analyze the weighted risk function in Eq. 23.

Following [8], we can use the following two lemmas for bounding the target domain risk. Lemma B.1 bounds the difference between the target risk and the weighted risk, and Lemma B.2 bounds the difference between the empirical and true weighted risks.

Lemma B.1 *Let h be a hypothesis in $\mathcal{H}$, then*

$$\left|\epsilon_\theta(h) - \epsilon_T(h)\right| \leq (1-\theta)\left(\frac{1}{2} d_{\mathcal{H}\triangle\mathcal{H}}(\mathcal{D}_S, \mathcal{D}_T) + \lambda\right) \tag{24}$$

Lemma B.2 *Let $\mathcal{H}$ be a hypothesis space of VC-dimension k. If we label βm points from $\mathcal{D}_T$ and $(1-\beta)m$ points from $\mathcal{D}_S$ by f_T and f_S respectively with $\beta = n_T/(n_S + n_T)$, then with probability at least $1-\delta$, for every $h \in \mathcal{H}$*

$$\left|\hat{\epsilon}_\theta(h) - \epsilon_\theta(h)\right| < \sqrt{\frac{\theta^2}{\beta} + \frac{(1-\theta)^2}{1-\beta}} \sqrt{\frac{k\log(2m) - \log\delta}{2m}} \tag{25}$$

Following [8], from these two lemmas, we can obtain the following theorem:

Theorem B.1 *Let $\mathcal{H}$ be a hypothesis space of VC-dimension k. Let $\mathcal{U}_S$ and $\mathcal{U}_T$ be unlabeled samples of size m' each, drawn from $\mathcal{D}_S$ and $\mathcal{D}_T$ respectively, and $\hat{d}_{\mathcal{H}\triangle\mathcal{H}}$ is the empirical distance between them. Let $\mathcal{L} = \mathcal{L}_S \bigcup \mathcal{L}_T$ be the labeled samples of size m generated by drawing $(1-\beta)m$ points from $\mathcal{D}_S$ and βm points from $\mathcal{D}_T$, labeling them according to f_S and f_T respectively. If $\hat{h} \in \mathcal{H}$ is the empirical minimizer of $\hat{\epsilon}_\theta(h)$ on $\mathcal{L}$, $h^*_T = min_{h\in\mathcal{H}}\ \epsilon_T(h)$*

is the target risk minimizer, and $\gamma = \min_{h\in\mathcal{H}} \epsilon_S(h) + \epsilon_T(h)$*, then with probability of at least* $1-\delta$*(over the choice of the samples)[8]*

$$\begin{aligned}\epsilon_T(\hat{h}) \le \epsilon_T(h_T^*) + 2\sqrt{\frac{\theta^2}{\beta} + \frac{(1-\theta)^2}{1-\beta}}\sqrt{\frac{k\log(2m)-\log\delta}{2m}} \\ +2(1-\theta)\left(\frac{1}{2}\hat{d}_{\mathcal{H}\triangle\mathcal{H}}(\mathcal{U}_S,\mathcal{U}_T) + 4\sqrt{\frac{2k\log(2m')+\log\left(\frac{4}{\delta}\right)}{m'}} + \gamma\right)\end{aligned} \tag{26}$$

Following [28], we can use the following theorem to bound the target risk $\epsilon_T(h)$.

Theorem B.2 *Let* $\mathcal{L}_T$ *be the labeled samples of size* βm *generated from* $\mathcal{D}_T$*, labeling them according to* f_T*, and e be the natural logarithm. For each ranking function* $h : \mathcal{X} \to \mathbb{R}$ *with zero training error, then with probability* $1-\delta$

$$\epsilon_T(h) \le \frac{2}{\beta m - 1}\left(k\log\left(\frac{8e(\beta m-1)}{k}\right)\log\left(32(\beta m-1)\right) + \log\left(\frac{8(\beta m-1)}{\delta}\right)\right)$$

By plugging theorem B.2 into theorem B.1, we can obtain the generalization bound for our HCD Ranking problem as show in theorem 3.1. □

References

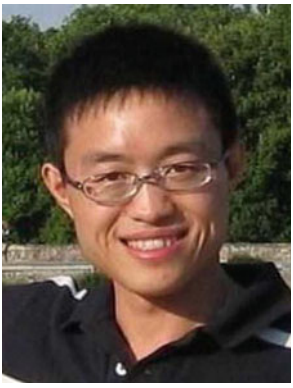
1. Agarwal A, Chakrabarti S, Aggarwal S (2006) Learning to rank networked entities. In: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD'06), pp 14–23
2. Amini M-R, Truong T-V, Goutte C (2008) A boosting algorithm for learning bipartite ranking functions with partially labeled data. In: Proceedings of the 31st annual international ACM SIGIR conference on research and development in information retrieval (SIGIR'08), pp 99–106
3. Argyriou A, Evgeniou T, Pontil M (2006) Multi-task feature learning. In: Proceedings of the 18th neural information processing systems (NIPS'06), pp 41–48
4. Baccini A, Dejean S, Lafage L, Mothe J (2011) How many performance measures to evaluate information retrieval systems? Knowl Inf Syst 1–21. doi:10.1007/s10115-011-0391-7
5. Baeza-Yates R, Ribeiro-Neto B (1999) Modern information retrieval. ACM Press, New York
6. Bar-Yossef Z, Guy I, Lempel R, Maarek YS, Soroka V (2008) Cluster ranking with an application to mining mailbox networks. Knowl Inf Syst 14(1):101–139
7. Bickel S, Brückner M, Scheffer T (2007) Discriminative learning for differing training and test distributions. In: Proceedings of the 24th international conference on machine learning (ICML'07), pp 81–88
8. Blitzer J, Crammer K, Kulesza A, Pereira F, Wortman J (2007) Learning bounds for domain adaptation. In: Proceedings of the 19th neural information processing systems (NIPS'07), pp 129–136
9. Blitzer J, McDonald R, Pereira F (2006) Domain adaptation with structural correspondence learning. In: Proceedings of conference on empirical methods in natural language processing (EMNLP'06), pp 120–128
10. Bonilla E, Chai KM, ChrisWilliams (2008) Multi-task gaussian process prediction. In: Proceedings of the 20th neural information processing systems (NIPS'08), pp 153–160
11. Brefeld U, Scheffer T (2005) Auc maximizing support vector learning. In: Proceedings of the 2nd workshop on ROC analysis in machine learning (ROCML 2005)
12. Buckley C, Voorhees EM (2004) Retrieval evaluation with incomplete information. In: Proceedings of the 27th annual international ACM SIGIR conference on research and development in information retrieval (SIGIR'04), pp 25–32
13. Burges C, Shaked T, Renshaw E, Lazier A, Deeds M, Hamilton N, Hullender G (2005) Learning to rank using gradient descent. In: Proceedings of the 22th international conference on machine learning (ICML'05), pp 89–96
14. Chapelle O, Shivaswamy P, Vadrevu S, Weinberger K, Zhang Y, Tseng B (2010) Multi-task learning for boosting with application to web search ranking. In: Proceedings of the 16th ACM SIGKDD international conference on knowledge discovery and data mining (KDD'10), pp 1189–1198
15. Chen K, Lu R, Wong CK, Sun G, Heck L, Tseng B (2008) Trada: tree based ranking function adaptation. In: Proceedings of the 17th ACM international conference on information and knowledge management (CIKM'08), pp 1143–1152

16. Cui J, Liu H, He J, Li P, Du X, Wang P (2011) Tagclus: a random walk-based method for tag clustering. Knowl and Inf Syst 27(2):193–225
17. Czarnowski I (2011) Cluster-based instance selection for machine classification. Knowl Inf Syst
18. Dai W, Jin O, Xue G, Yang Q, Yu Y (2009) Eigentransfer: a unified framework for transfer learning. In: Proceedings of the 26th annual international conference on machine learning (ICML'09), pp 193–200
19. Dai W, Yang Q, Xue G-R, Yu Y (2007) Boosting for transfer learning. In: Proceedings of the 24th international conference on machine learning (ICML'07), pp 193–200
20. Duh K, Kirchhoff K (2008) Learning to rank with partially-labeled data. In: Proceedings of the 31st annual international ACM SIGIR conference on research and development in information retrieval (SIGIR'08), pp 251–258
21. Evgeniou T, Pontil M (2004) Regularized multi-task learning. In: Proceedings of the tenth ACM SIGKDD international conference on knowledge discovery and data mining (KDD'04), pp 109–117
22. Gao J, Fan W, Jian J, Han J (2008) Knowledge transfer via multiple model local structure mapping. In: Proceeding of the 14th ACM SIGKDD international conference on knowledge discovery and data mining (KDD'08), pp 283–291
23. Gao J, Fan W, Sun Y, Han J (2009) Heterogeneous source consensus learning via decision propagation and negotiation. In: Proceedings of the 15th ACM SIGKDD international conference on knowledge discovery and data mining(KDD'09), pp 339–348
24. Gao J, Wu Q, Burges C, Svore K, Su Y, Khan N, Shah S, Zhou H (2009) Model adaptation via model interpolation and boosting for web search ranking. In: Proceedings of the 2009 conference on empirical methods in natural language processing (EMNLP'09), pp 505–513
25. Geng B, Yang L, Xu C, Hua X (2009) Ranking model adaptation for domain-specific search. In: Proceeding of the 18th ACM conference on information and knowledge management (CIKM'09), pp 197–206
26. Gupta SK, Phung D, Adams B, Tran T, Venkatesh S (2010) Nonnegative shared subspace learning and its application to social media retrieval. In: Proceedings of the 16th ACM SIGKDD international conference on knowledge discovery and data mining (KDD'10), pp 1169–1178
27. He J, Liu Y, Lawrence R (2009) Graph-based transfer learning. In: Proceeding of the 18th ACM conference on information and knowledge management (CIKM'09), pp 937–946
28. Herbrich R, Graepel T, Obermayer K (2000) Large margin rank boundaries for ordinal regression. MIT Press, Cambridge
29. Hoi SC, Jin R (2008) Semi-supervised ensemble ranking. In: Proceedings of association for the advancement of artificial intelligence (AAAI'08), pp 634–639
30. Jarvelin K, Kekalainen J (2000) Ir evaluation methods for retrieving highly relevant documents. In: Proceedings of the 23th annual international ACM SIGIR conference on research and development in information retrieval (SIGIR'00), pp 41–48
31. Jebara T (2004) Multi-task feature and kernel selection for svms. In: Proceedings of the 21th international conference on machine learning (ICML'04), pp 55–62
32. Jiang L, Li C, Cai Z (2009) Learning decision tree for ranking. Knowl Inf Syst 20(1):123–135
33. Joachims T (2002) Learning to classify text using support vector machines. Dissertation
34. Joachims T (2006) Training linear svms in linear time. In: Proceedings of the 12th ACM SIGKDD international conference on knowledge discovery and data mining (KDD'06), pp 217–226
35. Kang U, Tsourakakis CE, Faloutsos C (2011) Pegasus: mining peta-scale graphs. Knowl Inf Syst 27(2):303–325
36. Lee S-I, Chatalbashev V, Vickrey D, Koller D (2007) Learning a meta-level prior for feature relevance from multiple related tasks. In: Proceedings of the 24th international conference on machine learning (ICML'07), pp 489–496
37. Li B, Yang Q, Xue X (2009) Transfer learning for collaborative filtering via a rating-matrix generative model. In: Proceedings of the 26th annual international conference on machine learning(ICML'09), pp 617–624
38. Ling X, Xue G, Dai W, Jiang Y, Yang Q, Yu Y (2008) Can chinese web pages be classified with english data source? In: Proceeding of the 17th international conference on World Wide Web (WWW'08), pp 969–978
39. Liu J, Ji S, Ye J (2009) Multi-task feature learning via efficient $l_{2,1}$-norm minimization. In: The twenty-fifth conference on uncertainty in artificial intelligence (UAI'09), pp 339–348
40. Liu T-Y, Xu J, Qin T, Xiong W, Li H (2007) Letor: Benchmark dataset for research on learning to rank for information retrieval. In: LR4IR 2007, in conjunction with SIGIR 2007
41. Mihalkova L, Mooney RJ (2009) Transfer learning from minimal target data by mapping across relational domains. In: Proceedings of the 21st international jont conference on artifical intelligence(IJCAI'09), pp 1163–1168

42. Pan SJ, Ni X, Sun J, Yang Q, Chen Z (2010) Cross-domain sentiment classification via spectral feature alignment. In: Proceedings of the 19th international World Wide Web conference(WWW'10), pp 751–760
43. Pan SJ, Yang Q (2010) A survey on transfer learning. IEEE Trans Knowl Data Eng (TKDE) 22(10): 1345–1359
44. Qin T, Liu T, Zhang X, Wang D, Xiong W, Li H (2008) Learning to rank relational objects and its application to web search. In: 17th international World Wide Web conference (WWW'08), pp 407–416
45. Raina R, Battle A, Lee H, Packer B, Ng AY (2007) Self-taught learning: Transfer learning from unlabeled data. In: Proceedings of the 24th international conference on machine learning (ICML'07), pp 759–766
46. Rosa KD, Metsis V, Athitsos V (2011) Boosted ranking models: a unifying framework for ranking prediction. Knowl Inf Syst 1–26. doi:10.1007/s10115-011-0390-8
47. Shi X, Liu Q, Fan W, Yu PS, Zhu R (2010) Transfer learning on heterogenous feature spaces via spectral transformation. In: Proceedings of the 2010 IEEE international conference on data mining (ICDM'10), pp 1049–1054
48. Szummer M, Jaakkola T (2002) Partially labeled classification with markov random walks. In: Advances in neural information processing systems (NIPS'02), pp 945–952
49. Tang J, Jin R, Zhang J (2008) A topic modeling approach and its integration into the random walk framework for academic search. In: Proceedings of 2008 IEEE international conference on data mining (ICDM'08), pp 1055–1060
50. Tang J, Zhang J, Yao L, Li J, Zhang L, Su Z (2008) Arnetminer: Extraction and mining of academic social networks. In: Proceedings of the 14th ACM SIGKDD international conference on knowledge discovery and data mining (SIGKDD'08), pp 990–998
51. Wall ME, Rechtsteiner A, Rocha LM (2003) Singular value decomposition and principal component analysis. Kluwer, Norwell 91–109
52. Wang B, Tang J, Fan W, Chen S, Yang Z, Liu Y (2009) Heterogeneous cross domain ranking in latent space. In: Proceedings of the eighteenth conference on information and knowledge management (CIKM'09), pp 987–996
53. Wang Z, Song Y, Zhang C (2008) Transferred dimensionality reduction. In: Machine learning and knowledge discovery in databases, European conference (ECML/PKDD'08), pp 550–565
54. Wong T-L, Lam W, Chen B (2009) Mining employment market via text block detection and adaptive cross-domain information extraction. In: Proceedings of the 32nd international ACM SIGIR conference on research and development in information retrieval(SIGIR'09), pp 283–290
55. Xie S, Fan W, Peng J, Verscheure O, Ren J (2009) Latent space domain transfer between high dimensional overlapping distributions. In: Proceedings of the 18th international conference on World wide web(WWW'09), pp 91–100
56. Xu J, Li H (2007) Adarank: a boosting algorithm for information retrieval. In: Proceedings of the 30th annual international ACM SIGIR conference on research and development in information retrieval (SIGIR'07), pp 391–398
57. Yang Q, Chen Y, Xue G, Dai W, Yu Y (2009) Heterogeneous transfer learning for image clustering via the social web. In: Proceedings of the joint conference of the 47th annual meeting of the ACL and the 4th international joint conference on natural language processing of the AFNLP: Volume 1 (ACL'09), pp 1–9
58. Yang Z, Tang J, Wang B, Guo J, Li J, Chen S (2009) Expert2bole: from expert finding to bole search. In: Proceeding of the 15th ACM SIGKDD international conference on knowledge discovery and data mining (KDD'09)
59. Yue Y, Finley T, Radlinski F, Joachims T (2007) A support vector method for optimizing average precision. In: Proceedings of the 30th annual international ACM SIGIR conference on research and development in information retrieval (SIGIR'07), pp 271–278
60. Zhai C, Lafferty J (2001) Model-based feedback in the language modeling approach to information retrieval. In: Proceedings of the 10th conference on information and knowledge management (CIKM'01), pp 403–410
61. Zheng Z, Chen K, Sun G, Zha H (2007) A regression framework for learning ranking functions using relative relevance judgments. In: Proceedings of the 30th annual international ACM SIGIR conference on research and development in information retrieval (SIGIR'07), pp 287–294
62. Zhong E, Fan W, Peng J, Zhang K, Ren J, Turaga D, Verscheure O (2009) Cross domain distribution adaptation via kernel mapping. In: Proceedings of the 15th ACM SIGKDD international conference on knowledge discovery and data mining(KDD'09), pp 1027–1036
63. Zhu J, Huang X, Song D, Ruger SM (2010) Integrating multiple document features in language models for expert finding. Knowl Inf Syst 23(1):29–54

Author Biographies

Bo Wang is a PhD candidate from the Department of Computer Science & Engineering, Nanjing University of Aeronautics and Astronautics (NUAA), China. He got his bachelor degree from NUAA. His research interests focus on data mining and information retrieval.

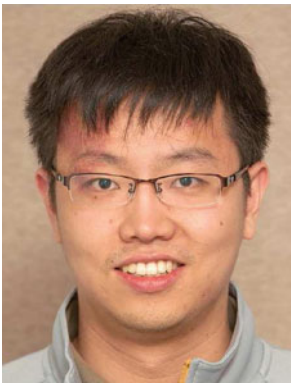


Jie Tang is an associate professor in Tsinghua University. His research interests are social network analysis, machine learning, and data mining.

Wei Fan (http://www.weifan.info) research interests are in risk analysis, high-performance computing, skewed distribution, cost-sensitive learning, streams, ensembles, graph mining, feature construction, feature selection, sample selection bias, transfer learning, novel applications, and commercial systems. He is particularly interested in simple, unconventional, but effective methods to solve difficult problems. He is on the editorial board of ACM TKDD. His thesis work on intrusion detection has been licensed by a start-up company. His co-teamed submission that uses Random Decision Tree (RDT) has won the ICDM'08 Contest championship. His coauthored paper in ICDM'06 that uses RDT won the best application paper award. The open source code of RDT is available from http://www.dice4dm.com. His coauthored paper in KDD'97 on distributed learning system won the runner-up best application paper award. He received 2010 IBM Outstanding Technical Achievement Award for contribution in building Infosphere Streams.

Songcan Chen received his B.S. degree in mathematics from Hangzhou University (now merged into Zhejiang University) in 1983. In 1985, he completed his M.S. degree in computer applications at Shanghai Jiaotong University and then worked at NUAA in January 1986. There he received a PhD degree in communication and information systems in 1997. Since 1998, as a full-time professor, he has been with Department of Computer Science & Engineering at NUAA. His research interests include pattern recognition, machine learning, and neural computing. In these fields, he has authored or coauthored over 140 scientific peer-reviewed papers and received Honorable Mentions of 2006 and 2008 Best Paper Awards of Pattern Recognition respectively.

Chenhao Tan is a PhD student from the Department of Computer Science, Cornell University. He is advised by Lillian Lee. He got his bachelor degree from Tsinghua University. His research interests focus on Social Network, Natural Language Processing, Machine Learning, and Algorithms.



Zi Yang is a PhD student in the School of Computer Science at Carnegie Mellon University. He is advised by Prof. Eric Nyberg. His research interests include Information Retrieval, Information Extraction, Social network Mining, and Machine Learning. He has a MS in Computer Science and Technology from Tsinghua University.