Pairwise Cross-Domain Factor Model for Heterogeneous Transfer Ranking

Bo Long
Yahoo! Labs
bolong@yahoo-inc.com

Yi Chang
Yahoo! Labs
yichang@yahoo-inc.com

Anlei Dong
Yahoo! Labs
anlei@yahoo-inc.com

Jianzhang He
Yahoo! Labs
jhe@yahoo-inc.com

ABSTRACT

Learning to rank arises in many information retrieval applications, ranging from Web search engine, online advertising to recommendation systems. Traditional ranking mainly focuses on one type of data source, and effective modeling relies on a sufficiently large number of labeled examples, which require expensive and time-consuming labeling process. However, in many real-world applications, ranking over multiple related heterogeneous domains becomes a common situation, where in some domains we may have a relatively large amount of training data while in some other domains we can only collect very little. Theretofore, how to leverage labeled information from related heterogeneous domain to improve ranking in a target domain has become a problem of great interests. In this paper, we propose a novel probabilistic model, pairwise cross-domain factor model, to address this problem. The proposed model learns latent factors(features) for multi-domain data in partially-overlapped heterogeneous feature spaces. It is capable of learning homogeneous feature correlation, heterogeneous feature correlation, and pairwise preference correlation for cross-domain knowledge transfer. We also derive two PCDF variations to address two important special cases. Under the PCDF model, we derive a stochastic gradient based algorithm, which facilitates distributed optimization and is flexible to adopt different loss functions and regularization functions to accommodate different data distributions . The extensive experiments on real world data sets demonstrate the effectiveness of the proposed model and algorithm.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; I.5.1 [**Pattern Recognition**]: Models-statistical

General Terms

algorithms

WSDM'12, February 8–12, 2012, Seattle, Washington, USA.
Copyright 2012 ACM 978-1-4503-0747-5/12/02 ...$10.00.

Keywords

Ranking, Homogeneous transfer ranking, Heterogeneous transfer ranking, Source Domain, Target Domain, Pairwise cross-domain factor model, Stochastic gradient descent.

1. INTRODUCTION

Ranking is the core component of many important information retrieval problems, such as web search, recommendation, computational advertising. Learning to rank represents an important class of supervised machine learning tasks with the goal of automatically constructing ranking functions from training data. As many other supervised machine learning problems, the quality of a ranking function is highly correlated with the amount of labeled data used to train the function. Due to the complexity of many ranking problems, a large amount of labeled training examples are usually required to learn a high quality ranking function. However, in general, it is very expensive and time-consuming to acquire labeled data.

On the other hand, in many real-world applications, ranking over multiple related domains becomes a common situation, where in some domains we may have a relatively large amount of training data while in some other domains we can only collect very little. In those situations, making use of labeled data from related domain to is a desirable direction to address the data scarcity in the target domain.

Besides ranking applications, this learning scenario is also popular for other applications and in general it has been studied as transfer learning in the literature. Existing transfer learning approaches mainly focus on knowledge transfer in the same feature space, i.e., the data from different domains are assumed in a common feature space (we refer to this scenario as *homogeneous transfer learning*). However, in practice, we often face the problem where the labeled data are scarce in their own feature space, whereas there may be a large amount of labeled heterogeneous data in another feature space. In fact, this problem arises frequently in today's information retrieval systems, such as search engines and recommendation systems. For example, a search engine system often conducts ranking learning tasks in various domains with different languages (e.g., English text search, Spanish text search, etc.), or different verticals/topics (e.g., news search, product search, etc.); here, data from an English language domain may be helpful for a Spanish language domain; however their data usually exist in different feature spaces which are language dependent. Similarly, different vertical search data could benefit each other with knowledge transfer; but they lie in different feature spaces. In such situations, it would be desirable to transfer the knowledge from heterogeneous domains to a target domain where we have relatively little training data available (we refer to this scenario as *heterogeneous transfer learning*). Note that unlike multiple view learning [7], there is no data instance correspondence between two domains in heterogeneous transfer learning.

For the homogeneous transfer learning, since the data are in a common feature space, the main challenge is to overcome the data distribution difference to learn domain correlation for knowledge transfer. On the other hand, for the heterogeneous transfer learning, the domain difference is beyond distribution difference, since distributions from heterogeneous spaces are not even comparable. Therefore, in general heterogeneous transfer learning is more challenging.

When it comes to ranking, the problem becomes *heterogeneous transfer ranking*, which is even more challenging due to the following facts. First, unlike in classification or regression, in ranking the labels (relevance scores) for different domains may not be comparable. For example, a domain can have five grade relevance scores; another domain can have binary relevance scores. In fact, since the relevance scores may be query-dependent, the absolute values are not important and the preference order between instances are more important. Therefore, heterogeneous transfer ranking needs to catch correlations between preference orders from different domains, instead the traditional label correlations in classification and regression. Second, in general, a ranking application needs thousands of (or millions of) training examples. It is important to develop the method that can scale well to large data sets.

In this paper, we propose a general probabilistic model, *Pairwise Cross-Domain Factor* (PCDF) model, for heterogeneous transfer ranking. PCDF model is a feature-based transfer ranking model that learns common latent factors (features) to transfer knowledge cross multiple heterogeneous ranking domains. PCDF assumes that (1) homogeneous features, heterogeneous features, and hidden relevance scores are generated conditioning on latent factors and (2) preference orders are generated conditioning on hidden relevance scores. Through direct and indirect parameter sharing for the generative processes in different domains, the latent factors catch different types of domain correlations to extract the common knowledge for different domains. Our contributions can be summarized as follows.

- We introduce the concepts of *two-component latent factor* and *hidden relevance score* as key concepts to model the the feature heterogeneity and preference pair in heterogeneous transfer ranking;

- We propose a novel heterogeneous transfer learning model, PCDF model, which is capable of learning homogeneous feature correlation, heterogeneous feature correlation, and preference order correlation. PCDF is applicable to various cross-domina ranking applications with different data distribution assumptions.

- We also derive two variations of the PCDF model to address two important special cases, which provide a new homogeneous transfer ranking model and a general transfer learning model beyond ranking applications.

- Under PCDF model, we derive an efficient stochastic gradient descent algorithm that is ready for distributed computation and is flexible to adopt different loss functions and regularization functions to accommodate different data distributions.

2. RELATED WORK

Transfer ranking is an overlapping field of , transfer learning, and learning to rank.

2.1 Transfer Learning

Transfer learning approaches can be mainly categorized into three classes.

A popular class of transfer learning methods is instance-based [4, 15, 31, 5, 25, 17, 35], which assumes that certain parts of the data in the source domain can be reused for the target domain by re-weighting. [26] proposed a heuristic method to remove “misleading” training instances from source domain so as to include “good” instances from labeled source-domain instances and unlabeled target-domain instances. [15] introduced a boosting algorithm, TrAdaBoost, which assumes that the source and target domain data use exactly the same set of features and labels, but the distributions of the data in the two domains are different. TrAdaBoost attempts to iteratively re-weight the source domain data and target domain data to reduce the effect of the “bad” source data while encourage the “good” source data to contribute more for the target domains. [4] proposed a framework to simultaneously re-weight the source domain data and train models on the re-weighted data with a kernel logistic regression classifier.

Another category of approaches can be viewed as model-based approaches [34, 28, 18, 8], which assumes that the source tasks and the target tasks share some parameters or priors of their models. An efficient algorithm MT-IVM [28], which is based on Gaussian Process (GP), was proposed to handle multi-domain learning case. MT-IVM tries to learn parameters of GP over multiple tasks by assigning the same GP prior to the tasks. Similarly, Hierarchical Bayes (HB) is used with GP for multi-task learning [34]. [18] borrowed the idea of [34] and used SVMs for multi-domain learning. The parameters of SVMs for each domain is assumed to be separable into two terms: a common term across tasks and a task specific term. [32] proposed a consensus regularization framework for transfer learning from multiple source domains to a target domain.

The third category of transfer learning approaches are feature based. [6, 33, 14, 1, 2, 3, 30], where a feature representation is learned for the target domain and used to transfer knowledge across domains. A structural correspondence learning (SCL) algorithm [6] is proposed to use unlabeled data from the target domain to extract features so as to reduce the difference between source and target domains. A simple kernel mapping function is introduced in [16], which maps the data from both domains to a high-dimensional feature space. [33] proposed to apply sparse coding, an unsupervised feature construction method, to learning higher level features across domain. On the other hand, heterogeneous transfer learning starts to attract attention very recently. We notice that [39] extends PLSA to a specific application, using social Web data to help image clustering; [37] proposes a manifold alignment based approach for heterogeneous domain adaptation; [24] formulates heterogeneous transfer learning as multi-task and multi-view learning and proposes a graph-based solution; [23] focus on single task learning with multiple outlooks, which is also related to heterogeneous transfer learning.

2.2 Learning to Rank

In recent years, the ranking problem is frequently formulated as a supervised machine learning problem [27, 9, 19, 38, 13, 41, 22]. These learning-to-rank approaches are capable of combining different kinds of features to train ranking functions. The problem of ranking can be formulated as that of learning a ranking function from pair-wise preference data. The idea is to minimize the number of contradicting pairs in training data. For example, RankSVM [27] uses support vector machines to learn a ranking function from preference data. RankNet [9] applies neural network and gradient descent to obtain a ranking function. RankBoost [19] applies the idea of boosting to construct an efficient ranking function from a set of weak ranking functions. The studies reported in [41] proposes a framework called GBRank using gradient descent in function spaces, which is able to learn relative ranking information in the context of web search. [10] proposes a new probabilistic method for listwise ranking. Specifically it introduces two probability models, respectively referred to as permutation probability and top k probability, to define a listwise loss function for learning.

A few studies have been applied the idea of transfer learning for learning to rank problem. Zha et al.[40] uses multitask learning technique to incorporate query difference, where each query is regarded as a task. However, the objective of this work is to learn a single ranking function instead of multiple functions for multiple tasks. TransRank[11] considers cross-domain information to attack transfer learning problem for ranking, which utilizes the labeled data from a source domain to enhance the learning of ranking function in the target domain with augmented features. However, this approach does not make use of unlabeled data. Gao et al.[21] explore several model adaptation methods for Web search ranking. They trained two ranking functions separately, then interpolated the two functions for the final result, and their experiments show that the simple model interpolation method achieves the best results. Similarly, heterogeneous transfer ranking is rarely touched in the literature. We notice that [36] proposes a regularized framework to addresses ranking cross heterogeneous domains. It simultaneously minimize two loss functions corresponding to two related domains by mapping each domain onto a shared latent space.

Among those transfer learning approaches, in general instance-based and model-based approaches depend on the assumption of homogeneous feature more strongly than feature-based approaches. Another advantage of feature-based approaches is its flexibility of adopting different base ranking learners in real applications, i.e., after the common latent features learnt from different domains, it is flexible to use any ranking learner on the new training data with common latent features to train ranking functions. Those motivate us to focus on deriving a feature-based model in this study.

3. PCDF MODEL

3.1 Problem Formulation

For ease of exposition and to avoid notational clutter, we use the terms, *target domain* and *source domain*, to distinguish two given domains in a transfer learning task, though the discussions in this study are applicable to the situation that two domains are exchangeable and mutually helpful and can also be easily extended to multiple domains.

We begin with notations. We consider that the target domain data exist in a $d_t + d_c$ dimension space and the source domain data exist in a $d_s + d_c$ dimension space, where d_c is the number of dimensions for their overlapped feature space (denoted as $\mathcal{S}^c$); d_t and d_s are the number of dimensions for their dedicated feature spaces (denoted as $\mathcal{S}^t$ and $\mathcal{S}^d$), respectively. For traditional homogeneous transfer learning , all data are in the same feature space, i.e., $d_t = 0$ and $d_s = 0$. For totally heterogeneous transfer learning, the feature spaces for the different domains has no overlapping, i.e., $d_c = 0$. In this study, we consider the most general case, partially overlapped heterogeneous feature spaces, which arises frequently in real applications.

We let n_t and n_s denote the numbers of instances in the target domain and the source domain, respectively. We let $X^{(t)} = [X^{(td)}X^{(tc)}]$ denote the target domain data, where $X^{(td)} \in \mathbb{R}^{n_t \times d_t}$ denote the target domain data in its dedicated feature space, and $X^{(tc)} \in \mathbb{R}^{n_t \times d_c}$ denote target domain data in the common feature space. Similarly, $X^{(s)} = [X^{(sd)}X^{(sc)}]$ denotes the source domain data, where $X^{(sd)} \in \mathbb{R}^{n_s \times d_s}$ denote the source domain data in its dedicated feature space, and $X^{(sc)} \in \mathbb{R}^{n_s \times d_c}$ denote the source domain data in the overlapped feature space. To denote the ith data instance in the target domain or source domain, we use $X^{(t)}_{i\cdot}$ or $X^{(s)}_{i\cdot}$.

Furthermore, we let $R^{(t)}_{ij}$ denote the preference value between ith instance and jth instance in the target domain such that

$$R^{(t)}_{ij} \begin{cases} > 0 & \text{if } i\text{th instance is preferred over } j\text{th instance,} \\ = 0 & \text{if } i\text{th and } j\text{th instance are equally preferred,} \\ < 0 & \text{if } j\text{th instance is preferred over } i\text{th instance.} \end{cases} \tag{1}$$

where $1 \leq i, j \leq n_t$ (however it is not necessary that there is $R^{(t)}_{ij}$ for any pair of i and j in the data). In general $R^{(t)}_{ij} \in \mathbb{R}$. In the special case that only the preference order matters, $R^{(t)}_{ij} \in \{-1, 0, +1\}$. Note that even the data are given with relevance labels or lists of ordered instances, they can be easily converted to the pairwise preferences. Similarly, we let $R^{(s)}_{ij}$ denote the preference value between ith instance and jth instance in the source domain.

In heterogeneous transfer ranking, given target domain data, $X^{(td)}$, $X^{(tc)}$, and $R^{(t)}$, and source domain data, $X^{(sd)}$, $X^{(sc)}$, and $R^{(s)}$, they are three types of domain difference to impede us from directly applying target domain data to the target domain: different feature distributions in the shared feature space; different dedicated feature spaces; different distributions for pairwise preference values.

On the other hand, we need to catch three types of domain correlations for knowledge transfer. The first one is homogeneous feature correlation hidden in the overlapped feature space, i.e., correlation between $X^{(tc)}$ and $X^{(sc)}$, which is the focus of traditional homogeneous transfer learning. The second one is heterogeneous feature correlation hidden in the dedicated feature spaces, i.e., correlation between $X^{(td)}$ and $X^{(sd)}$. The third one is preference correlation, i.e., the correlation between $R^{(t)}$ and $R^{(s)}$.

3.2 Model Formulation

In this study, we consider a generative model, in which features and pairwise preferences are generated conditioning on the latent variables.

3.2.1 Feature Generating

Fist we assume that the two domains' features are generated conditioning on the common latent factors with maximum domain correlations and minimum domain differences. In heterogenous transfer ranking, there are two type of feature correlations between the two domains and intuitively it is difficult to catch them in one type of latent factor. Base on this observation, we propose of the concept, two-component latent factor. The two-component latent factors for the two domains are given as follows,

$$Z^{(t)} = [Z^{(td)} Z^{(tc)}] \tag{2}$$

and

$$Z^{(s)} = [Z^{(sd)} Z^{(sc)}], \tag{3}$$

where $Z^{(t)} \in \mathbb{R}^{n_t \times (k_d + k_c)}$, $Z^{(td)} \in \mathbb{R}^{n_d \times k_c}$, $Z^{(tc)} \in \mathbb{R}^{n_t \times k_c}$, $Z^{(s)} \in \mathbb{R}^{n_s \times (k_c + k_d)}$, $Z^{(sd)} \in \mathbb{R}^{n_s \times k_d}$, $Z^{(sc)} \in \mathbb{R}^{n_s \times k_c}$, k_c is the dimension of latent factors for the common features, and k_d is the dimensions of latent factors for the dedicated features. In the target domain latent factor $Z^{(t)}$, the component $Z^{(td)}$ is to catch heterogeneous feature correlation and the component $Z^{(tc)}$ is to catch homogeneous feature correlation.

Then, the common features in the overlapped feature space are generated according to the following probabilities,

$$X^{(tc)} \sim p(X^{(tc)} | f(Z^{(tc)}; P^{(c)})) \tag{4}$$

$$X^{(sc)} \sim p(X^{(sc)} | f(Z^{(sc)}; P^{(c)})), \tag{5}$$

where $f(\cdot)$ is a link function and $P^{(c)}$ is the function parameter. Through the shared parameter $P^{(c)}$, the latent factors $Z^{(tc)}$ and $Z^{(sc)}$ catch common knowledge from the two domains' common features.

However, for the dedicated features $X^{(td)}$ and $X^{(sd)}$, since they are in the different feature spaces, in general direct knowledge transfer by sharing the link function parameter is not feasible. For example, if $X^{(td)}$ and $X^{(sd)}$ have different dimensions and we use the popular linear link function such that $f(Z; P) = ZP$, then it is not feasible for a shared parameter matrix $P^{(c)}$ to make the latent factor $Z^{(td)}$ and $Z^{(sd)}$ have the same dimension k_d. In other words, the parameter sharing is a too strong assumption for the heterogeneous features. On the other hand, it is more reasonable to learn the heterogeneous feature correlation indirectly through the interaction between different types of latent factors and interaction between latent factors and pairwise preferences (we discuss more details later). Therefore, we assume that the dedicated features are generated with their own link function parameters as follows,

$$X^{(td)} \sim p(X^{(td)} | g(Z^{(td)}; P^{(td)})) \tag{6}$$

$$X^{(sd)} \sim p(X^{(sd)} | g(Z^{(sd)}; P^{(sd)})). \tag{7}$$

where $g(\cdot)$ is a link function and $P^{(td)}$ and $P^{(sd)}$ are the function parameters.

Furthermore, we assume prior distributions for $P^{(c)}$, $P^{(td)}$, and $P^{(sd)}$ to reduce over fitting,

$$P^{(c)} \sim p(P^{(c)}; \lambda_c), \tag{8}$$

$$P^{(td)} \sim p(P^{(td)}; \lambda_{td}), \tag{9}$$

$$P^{(sd)} \sim p(P^{(sd)}; \lambda_{sd}), \tag{10}$$

3.2.2 Pairwise Preferences Generating

To generate another observed variable, the pairwise preference score R_{ij}, we propose the concept of latent relevance score. We assume that each instance i has a latent relevance score y_i such that comparing a pair of y_i and y_j will give the pairwise preference between instance i and j.

In other words, we assume that Rij is generated conditioning on y_i and y_j for the both domains,

$$R^{(t)}_{ij} \sim p(R^{(t)}_{ij} | r(y^{(t)}_i, y^{(t)}_j)). \tag{11}$$

$$R^{(s)}_{ij} \sim p(R^{(s)}_{ij} | r(y^{(s)}_i, y^{(s)}_j)). \tag{12}$$

where r is a link function. An intuitive choice for r is the difference function, i.e., $r(a, b) = a - b$. For example, we can assume that the distribution of $\mathbb{R}_{ij}$ is the normal distribution with the difference of y_i and y_j as the mean such that

$$R_{ij} | y_i, y_j \sim \mathbf{N}(y_i - y_j, \sigma^2). \tag{13}$$

We further assume that the latent relevance score is generated conditioning on the latent factor.

$$y^{(t)} \sim p(y^{(t)} | h(Z^{(t)}; w)) \tag{14}$$

$$y^{(s)} \sim p(y^{(s)} | h(Z^{(s)}; w)), \tag{15}$$

where h(;) is a link function, $w \in \mathbb{R}^k$ is the function parameter, and $Z^{(t)}$ and $Z^{(s)}$ are defined as in (2) and (3). Therefore, through the latent relevance scores and the shared parameter w , the latent factor is able to catch the pairwise preference correlation in addition to homogeneous feature correlation and heterogeneous feature correlation.

Similarly, we assume prior distributions for $P^{(w)}$ to reduce over fitting,

$$w \sim p(w; \lambda_w), \tag{16}$$

3.2.3 PCDF and Its Variations

Figure 1 summarizes the PCDF mode as a Bayesian network. From Figure 1, we can observe that the two domains share two parameters $P^{(c)}$ and w, which are bridges of knowledge transfer. Through the information propagation among the Bayesian network, the two-component latent factors catch the three types of domain correlations, i.e., the common knowledge cross the two domains.

As a general model for heterogenous transfer ranking, PCDF can be easily modified to accommodate special cases.

First, it can be easily applied to homogeneous transfer ranking, i.e., two domains in a shared feature space. Figure 2 shows the PCDF model for homogenous data. From Figure 2, we observe that without the heterogeneous features and

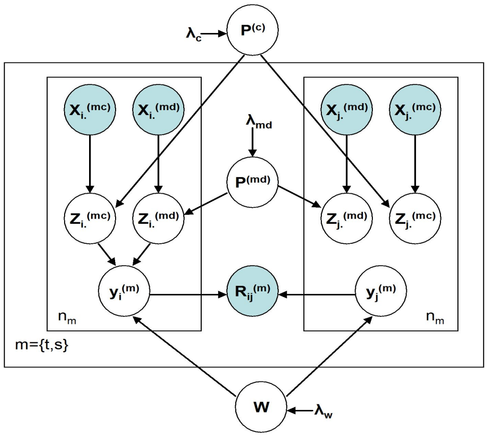

Figure 1: PCDF Bayesian network

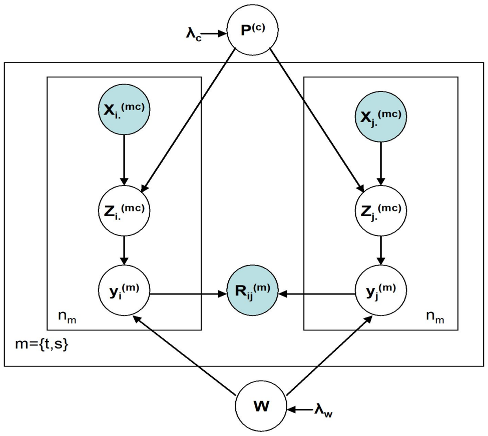

Figure 2: PCDF model for Homogeneous Data

heterogeneous latent factors, PCDF model is reduced to a new homogeneous transfer ranking model.

Second, with a little modification, the PCDF model can also be applied to the situation that data with absolute relevance scores. In such situation, $y^{(t)}$ and $y^{(s)}$ become observed variables and pairwise preferences are omitted. Hence, PCDF model is reduced to point-wise cross-domain factor model, which is shown in Figure 3. More interestingly, pointwise cross-domain factor model is beyond ranking applications, i.e., it can be directly applied to general regression and classification applications.

4. ALGORITHM DERIVATION

In this section, we derive the algorithm to learn the parameters for the PCDF model.

4.1 Objective Specification

The likelihood function of the PCDF model is given in Eq.(17), in which D denotes observed data; Ω denotes all parameters; Φ^m denotes the set of observed pairwise preferences of instances i and j for domain m.

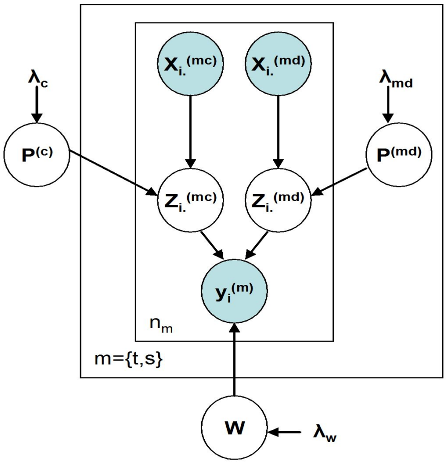

Figure 3: Pointwise Cross-domain Factor Model

$$
\begin{aligned}
p(D;\Omega) = \sum_{m=t,s} (\sum_{(i,j)\in\Phi^m} & (p(X_{i.}^{(mc)}|f(Z_{i.}^{(mc)};P^{(c)})) \\
& p(X_{i.}^{(md)}|g(Z_{i.}^{(md)};P^{(md)}))p(y_i^{(m)}|h(Z_{i.}^{(m)};w)) \\
& p(X_{j.}^{(mc)}|f(Z_{j.}^{(mc)};P^{(c)}))p(X_{j.}^{(md)}|g(Z_{j.}^{(md)};P^{(md)})) \\
& p(y_j^{(m)}|h(Z_{j.}^{(m)};w)))p(R_{ij}^{(m)}|r(y_i^{(m)},y_j^{(m)})) \\
& p(P^{(md)};\lambda_{md}))p(P^{(c)};\lambda_c)p(w;\lambda_w),
\end{aligned}
\tag{17}
$$

Eq. (17) is a general objective function. In this study, to instantiate Eq. (17), we adopt popular linear function for the link functions f,g, and h; we use intuitive difference function for r. Furthermore, It has been observed in the literature [12] that maximizing likelihood under a certain distribution corresponds to minimizing distance under the corresponding distortion measure. For example, the normal distribution, Bernoulli distribution, multinormial distribution and exponential distribution correspond to Euclidean distance, logistic loss, KL-divergence and Itakura-Satio distance, respectively. Therefore, the problem of minimizing the the negative log posterior of PCDF boils down to the following objective:

$$
\begin{aligned}
\min_{\Omega} \sum_{m=t,s} (\sum_{(i,j)\in\Phi^m} & (\alpha_{mc} l(X_{i.}^{(mc)}, Z_{i.}^{(mc)} P^{(c)})+ \\
& \alpha_{md} l(X_{i.}^{(md)}, Z_{i.}^{(md)} P^{(md)}) + \beta_m l(y_i^{(m)}, Z_{i.}^{(m)} w)+ \\
& \alpha_{mc} l(X_{j.}^{(mc)}, Z_{j.}^{(mc)} P^{(c)})+ \\
& \alpha_{md} l(X_{j.}^{(md)}, Z_{j.}^{(md)} P^{(md)}) + \beta_m l(y_j^{(m)}, Z_{j.}^{(m)} w)+ \\
& \gamma_m l(R_{ij}^{(m)}, y_i^{(m)} - y_j^{(m)})) + \lambda_{md}\omega(P^{(md)})+ \\
& \lambda_c \omega(P^{(c)}) + \lambda_w \omega(w),
\end{aligned}
\tag{18}
$$

where α, β, γ, and λ are trade-off parameters; $l(.,.)$ are the loss functions (for convenience, we use l for all terms; in general, it could be different for different terms) corresponding to conditional distributions in Eq. (17); $\omega(.)$ is the regularization loss function corresponding to the prior distribution in Eq. (17).

The motivations for a computational framework instead of direct probabilistic inference are mainly two-fold: First, the two formulations are somewhat equivalent, i.e., the conditional distributions can be encoded precisely through the choice of loss functions; likewise, the prior distributions over parameters could also be readily translated into the regularization penalties. Secondly, computational models allow more scalable algorithms, e.g. via stochastic gradient descent, whereas probabilistic reasoning often requires Monte Carlo sampling or quite nontrivial variational approximations.

4.2 Optimization and Implementation

In general, minimizing (18) is a nonconvex problem regardless of the choice of the loss functions and regularizers. While there are convex reformulations for some settings, they tend to be computationally inefficient for large scale problems - the convex formulations require the manipulation of a full matrix which is impractical for anything beyond thousands of instances.

We established algorithms for distributed optimization based on the Hadoop MapReduce framework. The basic idea is to decompose the objective in (18) by optimizing with respect to each pairwise preference $R_{ij}^{(m)}$ and to combine the results for the parameters in the Reduce phase.

We briefly describe a stochastic gradient descent algorithm to solve the optimization of (18). The algorithm is computationally efficient and decouples different pairwise preferences. For a detailed discussion please see [42]. The algorithm loops over all the observations and updates the parameters by moving in the direction defined by negative gradient. Specifically, for each observation $R_{ij}^{(m)}$, the algorithm performs the following sequence of updating. First, the hidden variables related to instance i are updated as follows:

$$y_i^{(m)} = y_i^{(m)} - \delta(\beta_m l'(y_i^{(m)}, Z_{i.}^{(m)} w) + \gamma_m l'(R_{ij}^{(m)}, y_i^{(m)} - y_j^{(m)})); \tag{19}$$

$$\begin{aligned} Z_{i.}^{(md)} = Z_{i.}^{(md)} - \delta(&\alpha_{md} l'(X_{i.}^{(md)}, Z_{i.}^{(md)} P^{(md)}) \odot P^{(md)}\mathbf{1} + \\ &\beta_m l'(y_i^{(m)}, Z_{i.}^{(md)} w^{(d)}) \odot w^{(d)}), \end{aligned} \tag{20}$$

$$\begin{aligned} Z_{i.}^{(mc)} = Z_{i.}^{(mc)} - \delta(&\alpha_{mc} l'(X_{i.}^{(mc)}, Z_{i.}^{(mc)} P^{(c)}) \odot P^{(c)}\mathbf{1} + \\ &\beta_m l'(y_i^{(m)}, Z_{i.}^{(mc)} w^{(c)}) \odot w^{(c)}), \end{aligned} \tag{21}$$

where δ is the learning rate, $\odot$ denotes elementwise multiplication, $\mathbf{1}$ denotes vector of 1's, and for convenience we write w into $\left[\begin{smallmatrix} w^{(d)} \\ w^{(c)} \end{smallmatrix} \right]$.

Second, the latent variables related to instance j are updated as follows:

$$y_j^{(m)} = y_j^{(m)} - \delta(\beta_m l'(y_j^{(m)}, Z_{j.}^{(m)} w) - \gamma_m l'(R_{ij}^{(m)}, y_i^{(m)} - y_j^{(m)})); \tag{22}$$

$$\begin{aligned} Z_{j.}^{(md)} = Z_{j.}^{(md)} - \delta(&\alpha_{md} l'(X_{j.}^{(md)}, Z_{j.}^{(md)} P^{(md)}) \odot P^{(md)}\mathbf{1} + \\ &\beta_m l'(y_j^{(m)}, Z_{j.}^{(md)} w^{(d)}) \odot w^{(d)}), \end{aligned} \tag{23}$$

$$\begin{aligned} Z_{j.}^{(mc)} = Z_{j.}^{(mc)} - \delta(&\alpha_{mc} l'(X_{j.}^{(mc)}, Z_{j.}^{(mc)} P^{(c)}) \odot P^{(c)}\mathbf{1} + \\ &\beta_m l'(y_j^{(m)}, Z_{j.}^{(mc)} w^{(c)}) \odot w^{(c)}), \end{aligned} \tag{24}$$

Algorithm 1 General PCDF Algorithm

```
Input:{$X^{(t)}, R^{(t)}, X^{(s)}, R^{(s)}$} and an integer b (number of instances for batch updating parameters).
Output:{$P^{(td)}, P^{(sd)}, P^{(c)}, w$}.
Method:
Initialize {$Z^{(t)}, Z^{(s)}, y^{(t)}, y^{(s)}, P^{(td)}, P^{(sd)}, P^{(c)}, w$}
repeat
    for m=t,d do
        Randomly shuffle $R^{(m)}$
        Let $count = 0$
        for Each observed $R_{ij}^{(m)}$ do
            Let $count = count + 1$
            Perform updating rules (19)-(21).
            Perform updating rules (22)-(24)
            if count%b == 0 then
                Perform updating rules (25)-(27)
            end if
        end for
    end for
until convergence
```

Third, the parameters are updated as follows:

$$\begin{aligned} P^{(md)} = P^{(md)} - \delta(&\alpha_{md} l'(X^{(md)}, Z^{(md)} P^{(md)}) Z^{(md)} + \\ &\lambda_{md} \omega'(P^{(md)})), \end{aligned} \tag{25}$$

$$\begin{aligned} P^{(c)} = P^{(c)} - \delta(&\sum_{m=t,s} (\alpha_{mc} l'(X^{(mc)}, Z^{(mc)} P^{(c)}) Z^{(mc)}) + \\ &\lambda_c \omega'(P^{(c)})), \end{aligned} \tag{26}$$

$$\begin{aligned} w = w - \delta(&\sum_{m=t,s} (\beta_m l'(y^{(m)}, Z^{(m)} w) Z^{(m)}) + \\ &\lambda_w \omega'(w)). \end{aligned} \tag{27}$$

In summary, the algorithm loops over all the $R_{ij}^{(m)}$'s to perform updating rules (19)-(27) until it converges. In practice, the algorithm may not need to update parameters for each $R_{ij}^{(m)}$, since the changes of parameters may not be significant for one observation. Instead it could be more efficient to perform updating rules (25)-(27) after performing updating rules (19)-(24) on a batch of observations. General PCDF algorithm is summarized in Algorithm 1. Note that following the similar procedure, it is easy to derive the algorithms for the variations of the PCDF model in Figure 2 and Figure 3, PCDF for homogenous data and point-wise (regression-based) cross-domain factor model.

The proposed stochastic gradient descent based algorithm has two desirable properties. First, it is ready for distributed optimization based on the Hadoop MapReduce framework, since it decouples different pairwise preference observations. Second, it is flexible to adopt different loss functions and regularization function for different types of data with different distributions.

5. EXPERIMENTAL EVALUATION

As a general heterogeneous ranking algorithm, PCDF can be applied to different ranking applications with different data distributions. In this section, we apply PCDF to Web search data to demonstrate the properties and effectiveness of PCDF.

Although PCDF model is flexible to adopt different loss and regularization functions, in this study we evaluate PCDF with the most popular loss function, L2 loss, corresponding to normal distribution, i.e., we use L2 loss for all l in minimization (18). For the regularization functions, we evaluate both L2 loss (normal distribution prior) and L1 loss (laplace distribution prior). We denote those two algorithms as *PCDF-n-he* and *PCDF-l-he* (”n” is for normal distribution prior; ”l” is for laplace distribution prior; ”he” is for heterogeneous ranking).

To our best knowledge, there is no existing transfer learning algorithms directly applicable to partially overlapped heterogeneous feature spaces (However, we thank an anonymous reviewer for pointing out very recent works, like [37] and [24], which may be modified to this learning situation). In this study, we use two state-of-the-art homogeneous transfer learning algorithm, Sparse Coding (called *SC-ho*)[33, 29] and multi-task feature learning (called *MTFL-ho*)[2] as comparisons. Furthermore, an intuitive way for heterogeneous transfer learning is to treat one domain's dedicated features in another domain as missing values and apply existing homogeneous transfer learning to the data with missing values; hence we modify the sparse coding algorithm (it is difficult to modify MFLF for this) to handle missing values (called *SC-he*) to evaluate this idea. Finally, the baseline is using the original target domain training data only (called *TD*) for ranking learning.

We also evaluate algorithms for two PCDF variations in Figure 2 and Figure 3 to gain deep understanding of PCDF itself. Similarly we use L2 loss for all the loss functions; L2 and L1 loss for the regularizer functions. For PCDF for homogeneous data in Figure 2, they are denoted as *PCDF-n-ho* and *PCDF-l-ho*, respectively; for regression based cross-domain factor model in Figure 3, they are denoted as *RCDF-n-he* and *RCDF-l-he*.

In summary, we compare ten algorithms: TD, SC-ho, SC-he, MTFL-ho, PCDF-n-ho, PCDF-l-ho, RCDF-n-he, RCDF-n-he, PCDF-n-he, and PCDF-l-he.

5.1 Data

We use Web search data from a commercial search engine system, which conducts ranking learning tasks in various domains with different languages or different verticals. The source domain (denoted as S0) is general Web search for an English speaking country, which has relatively large amount of labeled data. The first target domain (denoted as T1) is general Web search for a Spanish speaking country; the second target domain (denoted as T2) is news article search for the same country as S0; the third target domain (denoted as T3) is answer search (providing text search for knowledge-sharing community) for another non-English speaking country.

In the data, each query-document example is represented by a feature vector. Those query-document examples in domain S and T1 are originally labeled using a five-grade labeling scheme and those in domain T1 and T2 are originally labeled using a four-grade labeling scheme. We then transform them into pairwise preference data.

The features generally fall into the following three categories. Query features comprise features dependent on the query only and have constant values across all the documents, for example, the number of terms in the query, whether or not the query is a person name, etc. Document features comprise features dependent on the document only and have constant values across all the queries, for example, the number of inbound links pointing to the document, the amount of anchor-texts in bytes for the document, and the language identity of the document, etc. Query-document features comprise features dependent on the relation of the query with respect to the document d, for example, the number of times each term in the query appears in the document d, the number of times each term in the query appears in the anchor-texts of the document, etc.

The four domains have an overlapped feature space consisting of text match features, which describes various aspects of text match between a query and a document, such as title match and body match. Each domain has its own dedicated feature space. For example, all domains have their own dedicated features related to term segmentation in a query which is language dependent; T2 has its dedicated features related to news articles's freshness. Table 1 summarizes the data information for four domains. In our experiments, 20% of data for each target domain are used as training data (i.e., labeled data are very scarce in target domains); 80% of them are used as test data to make evaluation robust; all the source domain data are used as training data to help the target domains.

5.2 Experimental Setting

To evaluate those transfer learning algorithms, first, the new latent features and parameters are learnt by the algorithms; then source domain and target domain training data with the new latent features then are used by a base ranking learner to train a ranking model; third, the ranking model are tested on the test data of the target domain, which are also projected into the new feature space.

For the base ranking learner, we use Gradient Boosting Decision Tree (GBDT)[20]. For performance measure of the ranking models, we adopt widely used Discounted Cumulative Gain (DCG). Specifically, we sue DCG-k, since users of a search engine are only interested in the top k results of a query rather than a sorted order of the entire document collection. In this study, we select k as 5 and 1. For every experimental setting, 10 runs are repeated and the average DCG of 10 runs is reported for each experiment setting.

The gradient of L1 regularization function is the discontinuous sign function. We approximate it by a steep soft sign function $l(x) = \frac{1-\exp(-\theta x)}{1+\exp(-\theta x)}$, where θ is a positive number controlling the ramp of $l(x)$ (we use $\theta = 100$).

For the dimension of the latent factor, in our preliminary experiments we observe that the performance is not sensitive to it as long as it is not too small. In this study, we simply set $k_c = 20$ and $k_d = 20$. Other parameters, such as the learning rate, are selected by cross-validation.

5.3 Results and Discussions

The relative weight of source domain data w.r.t the target domain data controls how much the source domain affect the target domain. It is very important, since it is directly affect the efficiency of the knowledge transfer. For example, the low weight will impede the knowledge transfer no matter how close the two domains are.

We evaluate the effects of relative weights of the source

Data set	Number of examples	Number of Preference Pairs	Overlapped Features	Dedicated Features
S0	50,121	325,684	67	121
T1	5,112	26,534	67	81
T2	5,087	25,332	67	93
T3	5,124	26,898	67	76

Table 1: Data summary for one source domain and three target domains.

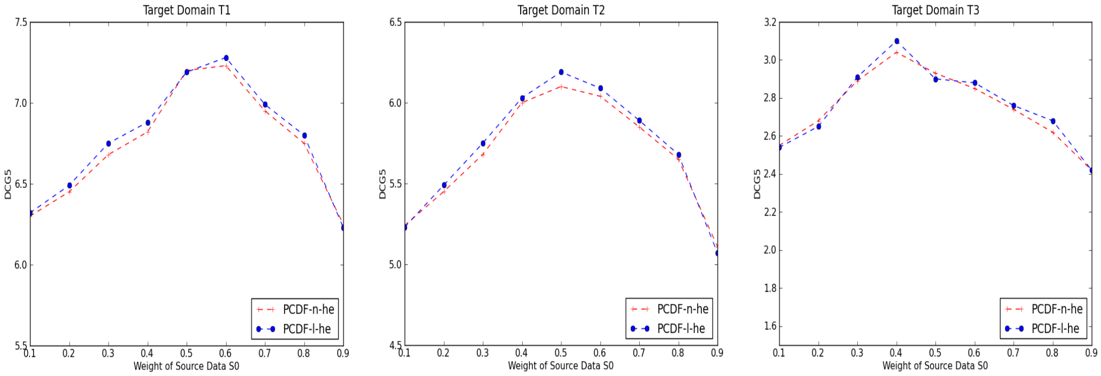

Figure 4: The effects of relative weights of source domain data on DCG5 performance.

domain data for PCDF-n-he and PCDF-l-he algorithms to illustrate this important aspect. Figure 4 shows The effects of relative weights of source domain data on DCG5 performance for all three target domains. From Figure 4, we observe that approximately the algorithms perform best at weight 0.6 for target domain T1, weight 0.5 for target domain T2, and weight 0.4 for target domain T3. The results implies that the T1 is the closest to the source domain, T2 is the second closest one, and T3 is the third closest one. This is consistent with our domain knowledge about the four domains. Note that it is flexible for PCDF algorithms to adopt the optimal relative weights that can be decided by cross-validation.

The DCG5 and DCG1 comparisons of the ten algorithms are shown in Figure 5 and Figure 6. From Figure 5 and Figure 6, we observe the following interesting facts:

- TD performs worst in all settings. This is due to insufficient training data for the target domains. This confirms the basic motivation of transfer ranking - to improve poor ranking performance in domains with insufficient training data.

- Overall PCDF-l-he performs best and PCDF-n-he performs second. This shows that PCDF can effectively catch domain correlations in both overlapped and heterogeneous feature spaces to improve learning in the target domain.

- PCDF-n-he and PCDF-l-he performs better than RCDF-n-he and RCDF-l-he. The possible reason is that PCDF catch correlations from preference orders, which matters more for ranking applications than absolute scores.

- PCDF-n-he and PCDF-l-he performs better than PCDF-n-ho and PCDF-l-ho, since PCDF-n-ho and PCDF-l-ho can use only features in the overlapped space and misses the knowledge transfer in the heterogenous feature spaces.

- PCDF-n-ho and PCDF-l-ho performs better than SC-ho and MTFL-ho. This shows that PCDF helps not only common knowledge learning cross heterogeneous feature spaces, but also in the same feature space, i.e., PCDF also provide an effective homogeneous transfer ranking model.

- MTFL algorithm performs better than SC algorithms. The possible reason for this is that as a supervised learning algorithm, MTFL learns the more informative latent features by making use of label information.

- SC-he does not perform better than SC-ho, even SC-he uses both overlapped features and heterogeneous features. This implies that simply treating one domain's dedicated features in another domain as missing values cannot effectively catch the the heterogeneous feature correlation.

- Overall, the L1 regularization algorithms performs better than L2 regularization algorithms. This shows that the choice of the regularization functions could have significant effects on the performance and hence, it is desirable for an algorithm to be flexible to adopt different regularization functions.

6. CONCLUSIONS

In this paper, we propose a novel probabilistic model, PCDF, for heterogeneous transfer ranking. The proposed model learns latent factors for multi-domain data in partially-overlapped heterogeneous feature spaces. It is capable of learning homogeneous feature correlation, heterogeneous feature correlation, and pairwise preference correlation for cross-domain knowledge transfer. We also derive two PCDF variations to address two important special cases. Under the PCDF model, we derive a stochastic gradient based algorithm, which facilitates distributed optimization and is flex-

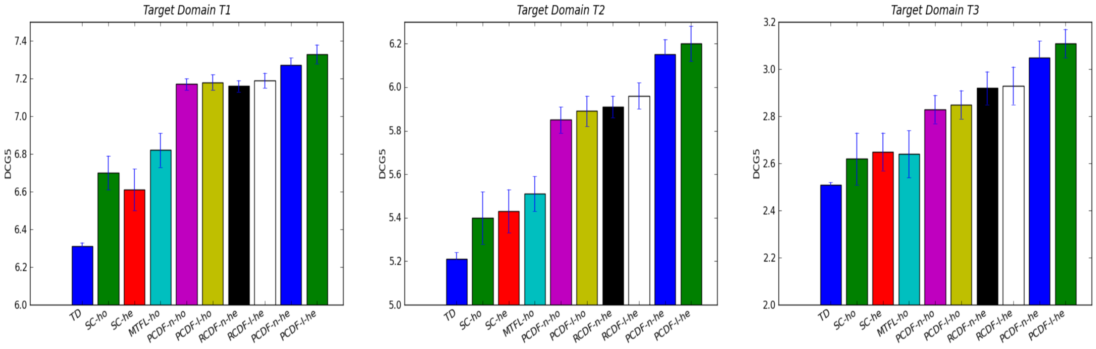

Figure 5: DCG5 comparisons of ten algorithms on the three target domains.

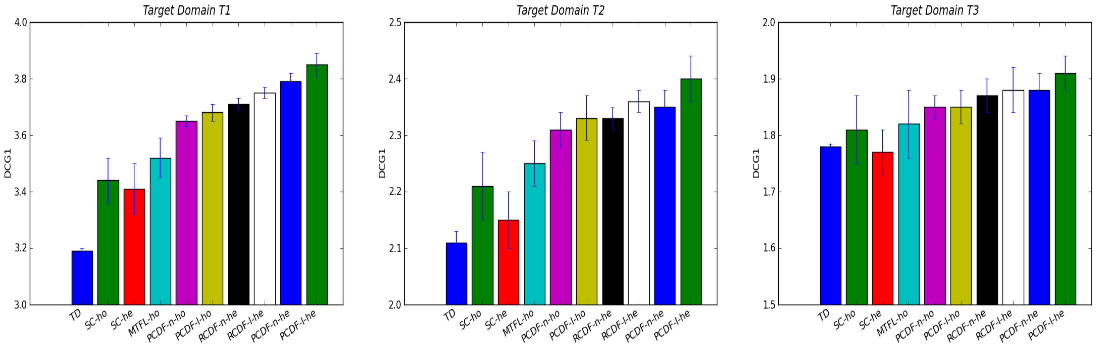

Figure 6: DCG1 comparisons of ten algorithms on the three target domains.

ible to adopt different loss functions and regularization functions to accommodate different data distributions . The extensive experiments on real Web search data sets demonstrate the effectiveness PCDF model and algorithms.

7. REFERENCES

[1] R. Ando and T. Zhang. A high-performance semi-supervised learning method for text chunking. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 1–9. Association for Computational Linguistics Morristown, NJ, USA, 2005.

[2] A. Argyriou, T. Evgeniou, and M. Pontil. Multi-task feature learning. In *Advances in Neural Information Processing Systems: Proceedings of the 2006 Conference*, page 41. MIT Press, 2007.

[3] A. Argyriou, C. Micchelli, M. Pontil, and Y. Ying. A spectral regularization framework for multi-task structure learning. *Advances in Neural Information Processing Systems*, 20, 2008.

[4] S. Bickel, M. Brückner, and T. Scheffer. Discriminative learning for differing training and test distributions. In *Proceedings of the 24th international conference on Machine learning*, pages 81–88. ACM New York, NY, USA, 2007.

[5] J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. Wortman. Learning bounds for domain adaptation. *Advances in Neural Information Processing Systems*, 20, 2008.

[6] J. Blitzer, R. McDonald, and F. Pereira. Domain adaptation with structural correspondence learning. In *Proceedings of the Empirical Methods in Natural Language Processing (EMNLP)*, 2006.

[7] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*, COLT' 98, pages 92–100, 1998.

[8] E. Bonilla, K. Chai, and C. Williams. Multi-task gaussian process prediction. *Advances in Neural Information Processing Systems*, 20:153–160.

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

[10] Z. Cao, T. Qin, T.-Y. Liu, M.-F. Tsai, and H. Li. Learning to rank: from pairwise approach to listwise approach. In *ICML '07*, pages 129–136, New York, NY, USA, 2007. ACM.

[11] D. Chen, J. Yan, G. Wang, Y. Xiong, W. Fan, and Z. Chen. TransRank: A Novel Algorithm for Transfer of Rank Learning. In *IEEE ICDM Workshops*, 2008.

[12] M. Collins, S. Dasgupta, and R. Reina. A generalizaionof principal component analysis to the exponential family. In *NIPS'01*, 2001.

[13] C. Cortes, M. Mohri, and A. Rastogi. Magnitude-preserving ranking algorithms. In *Proceedings of the 24th ICML*, 2007.

[14] W. Dai, G. Xue, Q. Yang, and Y. Yu. Co-clustering based classification for out-of-domain documents. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 210–219. ACM New York, NY, USA, 2007.

[15] W. Dai, Q. Yang, G. Xue, and Y. Yu. Boosting for transfer learning. In *Proceedings of the 24th international conference on Machine learning*, pages 193–200. ACM New York, NY, USA, 2007.

[16] H. Daume. Frustratingly easy domain adaptation. In *Annual meeting-association for computational linguistics*, volume 45, page 256, 2007.

[17] H. Daume III and D. Marcu. Domain adaptation for statistical classifiers. *Journal of Artificial Intelligence Research*, 26:101–126, 2006.

[18] T. Evgeniou and M. Pontil. Regularized multi-task learning. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 109–117. ACM New York, NY, USA, 2004.

[19] Y. Freund, R. Iyer, R. Schapire, and Y. Singer. An efficient boosting algorithm for combining preferences. In *Proceedings of the Fifteenth International Conference on Machine Learning*, 1998.

[20] J. Friedman. Greedy function approximation: a gradient boosting machine. *Annals of Statistics*, pages 1189–1232, 2001.

[21] J. Gao, Q. Wu, C. Burges, K. Svore, Y. Su, N. Khan, S. Shah, and H. Zhou. Model adaptation via model interpolation and boosting for web search ranking. In *Proceedings of conference on Empirical Methods in Natural Language Processing*, 2009.

[22] J. Guiver and E. Snelson. Learning to rank with SoftRank and Gaussian processes. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, 2008.

[23] M. Harel and S. Mannor. Learning from multiple outlooks. In L. Getoor and T. Scheffer, editors, *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, ICML '11, pages 401–408, New York, NY, USA, June 2011. ACM.

[24] J. He and R. Lawrence. A graph-based framework for multi-task multi-view learning. In L. Getoor and T. Scheffer, editors, *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, ICML '11, pages 25–32, New York, NY, USA, June 2011. ACM.

[25] J. Huang, A. Smola, A. Gretton, K. Borgwardt, and B. Scholkopf. Correcting sample selection bias by unlabeled data. *Advances in neural information processing systems*, 19:601, 2007.

[26] J. Jiang and C. Zhai. Instance weighting for domain adaptation in NLP. In *Annual meeting-assosciation for computational linguistics*, volume 45, page 264, 2007.

[27] T. Joachims. Optimizing search engines using clickthrough data. In *Proceedings of ACM SIGKDD*, 2002.

[28] N. Lawrence and J. Platt. Learning to learn with the informative vector machine. In *Proceedings of the twenty-first international conference on Machine learning*. ACM New York, NY, USA, 2004.

[29] H. Lee, A. Battle, R. Raina, and A. Y. Ng. Efficient sparse coding algorithms. In *In NIPS*, pages 801–808. NIPS, 2007.

[30] S. Lee, V. Chatalbashev, D. Vickrey, and D. Koller. Learning a meta-level prior for feature relevance from multiple related tasks. In *Proceedings of the 24th international conference on Machine learning*, pages 489–496. ACM New York, NY, USA, 2007.

[31] X. Liao, Y. Xue, and L. Carin. Logistic regression with an auxiliary data source. In *MACHINE LEARNING-INTERNATIONAL WORKSHOP THEN CONFERENCE-*, volume 22, page 505, 2005.

[32] P. Luo, F. Zhuang, H. Xiong, Y. Xiong, and Q. He. Transfer learning from multiple source domains via consensus regularization. In *CIKM '08: Proceeding of the 17th ACM conference on Information and knowledge management*, pages 103–112, New York, NY, USA, 2008. ACM.

[33] R. Raina, A. Battle, H. Lee, B. Packer, and A. Ng. Self-taught learning: Transfer learning from unlabeled data. In *Proceedings of the 24th international conference on Machine learning*, pages 759–766. ACM New York, NY, USA, 2007.

[34] A. Schwaighofer, V. Tresp, and K. Yu. Learning Gaussian process kernels via hierarchical Bayes. *Advances in Neural Information Processing Systems*, 17:1209–1216, 2005.

[35] M. Sugiyama, S. Nakajima, H. Kashima, P. von Bunau, and M. Kawanabe. Direct importance estimation with model selection and its application to covariate shift adaptation. *Advances in Neural Information Processing Systems*, 20, 2008.

[36] B. Wang, J. Tang, W. Fan, S. Chen, Z. Yang, and Y. Liu. Heterogeneous cross domain ranking in latent space. In *Proceeding of the 18th ACM conference on Information and knowledge management*, CIKM '09, pages 987–996, 2009.

[37] C. Wang and S. Mahadevan. Heterogeneous domain adaptation using manifold alignment. In *IJCAI*, pages 1541–1546, 2011.

[38] J. Xu and H. Li. Adarank: a boosting algorithm for information retrieval. In *Proceedings of the 30th ACM SIGIR*, 2007.

[39] Q. Yang, Y. Chen, G.-R. Xue, W. Dai, and Y. Yu. Heterogeneous transfer learning for image clustering via the social web. ACL '09, pages 1–9, 2009.

[40] H. Zha, Z. Zheng, H. Fu, and G. Sun. Incorporating query difference for learning retrieval functions in world wide web search. In *Proceedings of the 15th ACM CIKM conference*, 2006.

[41] Z. Zheng, K. Chen, G. Sun, and H. Zha. A regression framework for learning ranking functions using relative relevance judgments. In *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 287–294, New York, NY, USA, 2007. ACM.

[42] M. Zinkevich, M. Weimer, A. Smola, and L. Li. Parallelized stochastic gradient descent. In J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 2595–2603, 2010.