Heterogeneous Domain Adaptation using Manifold Alignment

Chang Wang
IBM T. J. Watson Research Lab
19 Skyline Drive
Hawthorne, New York 10532
wangchan@us.ibm.com

Sridhar Mahadevan
Computer Science Department
University of Massachusetts
Amherst, Massachusetts 01003
mahadeva@cs.umass.edu

Abstract

We propose a manifold alignment based approach for heterogeneous domain adaptation. A key aspect of this approach is to construct mappings to link different feature spaces in order to transfer knowledge across domains. The new approach can reuse labeled data from multiple source domains in a target domain even in the case when the input domains do not share any common features or instances. As a pre-processing step, our approach can also be combined with existing domain adaptation approaches to learn a common feature space for all input domains. This paper extends existing manifold alignment approaches by making use of labels rather than correspondences to align the manifolds. This extension significantly broadens the application scope of manifold alignment, since the correspondence relationship required by existing alignment approaches is hard to obtain in many applications.

1 Introduction

Classification and ranking methods in machine learning usually rely on the availability of a large amount of labeled data to train a model. However, labeled data is often expensive to obtain. To save labeling effort, in many situations we want to transfer labeled information from one domain to another. This problem arises in a variety of applications in information retrieval, e-commerce, computer vision, and many other fields. To address this problem, the area of transfer learning in general, and domain adaptation in particular, has recently seen a surge of activity [Blitzer, McDonald, and Pereira, 2006; Daumé III and Marcu, 2006; Duan et al., 2009; Mansour, Mohri, and Rostamizadeh, 2009; Pan and Yang, 2010]. However, one limitation that has not been fully addressed is that most existing domain adaptation approaches assume that the source and target domains are defined by the same features, and the difference between domains primarily arises due to the difference between data distributions. This assumption is not valid in many scenarios such as cross-lingual retrieval and multimodal datasets involving words and images, where the source and target domains do not share common features.

In a general setting of the problem, we are given K related input domains: $X_1, \cdots, X_K$, where K can be larger than 2 and the K input domains do not have to share any common features or instances. The domain adaptation problem in this scenario is very challenging, especially when we consider the fact that the target domain only has very limited labeled information. This setting is in fact quite common in many real world applications, and one example is as follows: assume we have three collections of documents in English, Italian, and Arabic respectively. In these collections, there are sufficient labeled English and Italian documents, but few labeled Arabic documents. The task is to label the unlabeled Arabic documents, leveraging the labels from English and Italian collections. Most existing domain adaptation approaches [Blitzer, McDonald, and Pereira, 2006; Daumé III and Marcu, 2006; Duan et al., 2009] can not be directly applied to this setting, since the input domains are defined in different feature spaces. Approaches in transfer learning that heavily depend on labeled information (like [Dai et al., 2008]) might not work either, since no instance is shared across domains and the overfitting problem will arise when the amount of labeled information is very limited.

Recently, a new approach to transfer learning based on manifold alignment was proposed to address this problem [Ham, Lee, and Saul, 2005; Wang and Mahadevan, 2009]. The key idea underlying this approach is to map different domains to a new latent space, simultaneously matching the corresponding instances and preserving topology of each input domain. Manifold alignment makes use of both unlabeled and labeled data. The ability to exploit unlabeled data is particularly useful for domain adaptation, where the number of labeled instances in the target domain is usually limited. A key difficulty in applying manifold alignment to domain adaptation is that the alignment method requires specifying a small amount of cross-domain correspondence relationship to learn mapping functions, but such information may be difficult to obtain for most domain adaptation applications. In this paper, we extend the manifold alignment framework to domain adaptation by exploring how to use label information rather than correspondence to align input domains. This idea is based on the observation that many source and target domains defined by different features often share the same labels. Our approach is designed to learn mapping functions to project the source and target domains to a new latent space,

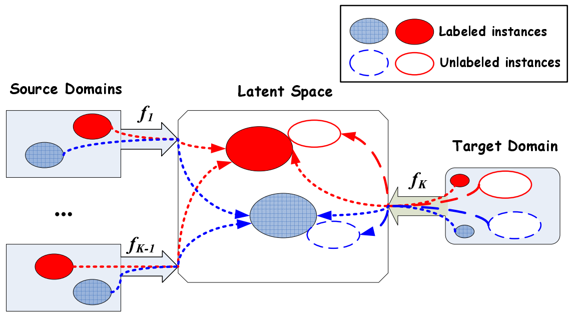

Figure 1: Illustration of manifold alignment using labels. Different colors represent different classes.

simultaneously matching the instances with the same labels, separating the instances with different labels and preserving the topology of each input domain.

The contributions of this paper are two-fold. From the perspective of domain adaptation, our contribution is a new approach to address the problem of transfer even in the case when the source and target domains do not share any common features or instances. It can also process multiple (> 2) input domains by exploring their common underlying structure. As a pre-processing step, our approach can be combined with existing domain adaptation approaches to learn a common feature space for all input domains. Our paper focuses on the construction of common latent space rather than studying which existing domain adaptation approach fits our framework the best. From the perspective of manifold alignment, our contribution is a new approach that uses labels rather than correspondences to learn alignment. This significantly broadens the application scope of manifold alignment. In experiments, we present case studies on how the new approach is applied to information retrieval.

2 Manifold Alignment using Labels

2.1 The Problem

Assuming we are given K input data sets, where the data instances come from c different classes. Let $X_k = (x_k^1, \cdots, x_k^{m_k})$ represent the k^{th} input data set, where the i^{th} instance x_k^i is defined by p_k features. X_k can be viewed as a matrix of size $p_k \times m_k$. The labels for the first l_k instances of X_k are given as $V_k = (v_k^1, \cdots, v_k^{l_k})$. When X_k corresponds to a source domain, l_k is usually large; when X_k corresponds to a target domain, l_k is usually small. In this problem formulation, $X_1, \cdots, X_K$ are assumed to be disjoint.

The problem is to construct K mapping functions, $f_1, \cdots, f_K$ to map the K input sets to a new d dimensional (latent) space, where (1) the topology of each set is preserved, (2) the instances from the same class (across the input sets) are mapped to similar locations and (3) the instances from different classes are well-separated from each other. An illustration of the problem is given in Figure 1.

2.2 High Level Explanation

We treat each input domain as a manifold. The goal is to construct K mapping functions to project the input domains to a new latent space preserving the topology of each domain, matching instances with the same labels and separating instances with different labels. To achieve this goal, we first create a matrix representation of the joint manifold modeling the union of all input domains. Each manifold is represented by a Laplacian matrix constructed from a graph defined by an "affinity" measure connecting nearby instances. The label information plays a key role in joining these adjacency graphs, forcing the instances with the same labels to be neighbors and separating instances with different labels. The joint manifold has features from all input domains, so its feature space is redundant. To remove the redundant features, we project the joint manifold to a lower dimensional space preserving manifold topology. This is a dimensionality reduction step, and is solved by a generalized eigenvalue decomposition. The resulting feature space is a common underlying space shared by all the input domains, and can be directly used for knowledge transfer across domains.

2.3 Notation

Before defining the cost function being optimized, we need to define some of the weight matrices used in the problem formulation. We first define the similarity matrix W_s, dissimilarity matrix W_d, their row sum matrices D_s, D_d and combinatorial Laplacian matrices L_s, L_d. Then we define matrices L and Z to model all the input domains.

♦ Similarity matrix $W_s = \begin{pmatrix} W_s^{1,1} & \cdots & W_s^{1,K} \\ \cdots & \cdots & \cdots \\ W_s^{K,1} & \cdots & W_s^{K,K} \end{pmatrix}$ is an $(m_1 + \cdots + m_K) \times (m_1 + \cdots + m_K)$ matrix, where $W_s^{a,b}$ is an $m_a \times m_b$ matrix. $W_s^{a,b}(i,j) = 1$, if x_a^i and x_b^j are from the same class; $W_s^{a,b}(i,j) = 0$, otherwise (including the case when the label information is not available). The corresponding diagonal row sum matrix is defined as $D_s(i,i) = \sum_j W_s(i,j)$, and the combinatorial graph Laplacian matrix $L_s = D_s - W_s$.

♦ Dissimilarity matrix $W_d = \begin{pmatrix} W_d^{1,1} & \cdots & W_d^{1,K} \\ \cdots & \cdots & \cdots \\ W_d^{K,1} & \cdots & W_d^{K,K} \end{pmatrix}$ is an $(m_1 + \cdots + m_K) \times (m_1 + \cdots + m_K)$ matrix, where $W_d^{a,b}$ is an $m_a \times m_b$ matrix. $W_d^{a,b}(i,j) = 1$, if x_a^i and x_b^j are from different classes; $W_d^{a,b}(i,j) = 0$, otherwise (including the case when the label information is not available). The corresponding diagonal row sum matrix is defined as $D_d(i,i) = \sum_j W_d(i,j)$, and the combinatorial Laplacian matrix $L_d = D_d - W_d$.

♦ To represent the topology of each given domain, we define W_k, D_k and L_k as follows. Let $W_k(i,j)$ represent the similarity of x_k^i and x_k^j. This similarity matrix can be computed as $e^{-\|x_k^i - x_k^j\|^2}$. We also define the corresponding diagonal row sum matrix D_k as $D_k(i,i) = \sum_j W_k(i,j)$ and combinatorial Laplacian matrix as $L_k = D_k - W_k$. Matrices

L and Z are defined as follows:

$$L = \begin{pmatrix} L_1 & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & \cdots & 0 & L_K \end{pmatrix} \text{ is an}$$

$(m_1 + \cdots + m_K) \times (m_1 + \cdots + m_K)$ matrix.

$$Z = \begin{pmatrix} X_1 & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & \cdots & 0 & X_K \end{pmatrix} \text{ is a}$$

$(p_1 + \cdots + p_K) \times (m_1 + \cdots + m_K)$ matrix.

2.4 The Cost Function

We then define A, B and C: three scalars to be used in the cost function.

$$A = 0.5 \sum_{a=1}^{K} \sum_{b=1}^{K} \sum_{i=1}^{m_a} \sum_{j=1}^{m_b} \|f_a^T x_a^i - f_b^T x_b^j\|^2 W_s^{a,b}(i,j),$$

If x_a^i and x_b^j are from the same class, but their embeddings are far away from each other, then A will be large. Minimizing A encourages the instances from the same class to be projected to similar locations in the new space.

$$B = 0.5 \sum_{a=1}^{K} \sum_{b=1}^{K} \sum_{i=1}^{m_a} \sum_{j=1}^{m_b} \|f_a^T x_a^i - f_b^T x_b^j\|^2 W_d^{a,b}(i,j),$$

If x_a^i and x_b^j are from different classes but their embeddings are close to each other in the new space, then B will be small. So maximizing B encourages the instances from different classes to be separated in the new space.

$$C = 0.5\mu \sum_{k=1}^{K} \sum_{i=1}^{m_k} \sum_{j=1}^{m_k} \|f_k^T x_k^i - f_k^T x_k^j\|^2 W_k(i,j).$$

If x_k^i and x_k^j are similar in their domain, then the corresponding $W_k(i,j)$ will be large. When the embeddings $f_k^T x_k^i$ and $f_k^T x_k^j$ are well-separated from each other in the new space, C becomes large. So minimizing C is to preserve topology of each given domain. μ is a weight parameter.

We want our algorithm to simultaneously achieve three goals in the new space: matching instances with the same labels, separating instances with different labels, and preserving topology of each given domain. So the overall cost function $\mathcal{C}(f_1, \cdots, f_K)$ to be minimized is:

$$\mathcal{C}(f_1, \cdots, f_K) = (A + C)/B.$$

We divide B rather than subtract B to save one parameter.

2.5 Theoretical Analysis

Let $\gamma = (f_1^T, \cdots, f_K^T)^T$ be a $(p_1 + \cdots + p_K) \times d$ matrix (representing K mapping functions) that we want to construct. The solution that minimizes the cost function is given in the following theorem.

Theorem 1. *The embedding that minimizes the cost function $\mathcal{C}(f_1, \cdots, f_K)$ is given by the eigenvectors corresponding to the smallest non-zero eigenvalues of the generalized eigenvalue decomposition $Z(\mu L + L_s)Z^T x = \lambda Z L_d Z^T x$.*

Proof. Given the input and the cost function, the problem is formalized as:

$$\begin{aligned} \{f_1, \cdots, f_K\} &= \text{argmin}_{f_1, \cdots, f_K}(\mathcal{C}(f_1, \cdots, f_K)) \\ &= \text{argmin}_{f_1, \cdots, f_K}(\frac{A + C}{B}) \end{aligned}$$

When $d = 1$, we can verify the following results hold:

$A = \gamma^T Z L_s Z^T \gamma, \ B = \gamma^T Z L_d Z^T \gamma, \ C = \gamma^T Z \mu L Z^T \gamma.$

$$\text{argmin}_{f_1, \cdots, f_K} \mathcal{C}(f_1, \cdots, f_K) = \text{argmin}_{f_1, \cdots, f_K} \frac{\gamma^T Z(L_s + \mu L)Z^T \gamma}{\gamma^T Z L_d Z^T \gamma}.$$

It follows directly from the Lagrange multiplier method that the optimal solution that minimizes the loss function $\mathcal{C}(f_1, \cdots, f_K)$ is given by the eigenvector corresponding to the minimum non-zero eigenvalue solution to the generalized eigenvalue problem:

$$Z(\mu L + L_s)Z^T x = \lambda Z L_d Z^T x.$$

When $d > 1$,

$$\begin{aligned} A &= Tr(\gamma^T Z L_s Z^T \gamma), B = Tr(\gamma^T Z L_d Z^T \gamma), \\ C &= Tr(\gamma^T Z \mu L Z^T \gamma). \end{aligned}$$

$$\begin{aligned} &\text{argmin}_{f_1, \cdots, f_K} \mathcal{C}(f_1, \cdots, f_K) \\ &= \text{argmin}_{f_1, \cdots, f_K} \frac{Tr(\gamma^T Z(L_s + \mu L)Z^T \gamma)}{Tr(\gamma^T Z L_d Z^T \gamma)}. \end{aligned}$$

Standard approaches [Wilks, 1963] show that the solution to $\gamma_1 \cdots \gamma_d$ that minimizes $\mathcal{C}(f_1, \cdots, f_K)$ is provided by the eigenvectors corresponding to the d lowest eigenvalues of the generalized eigenvalue decomposition equation:

$$Z(\mu L + L_s)Z^T x = \lambda Z L_d Z^T x.$$

□

The mapping functions $f_1, \cdots, f_K$ are linear. In some scenarios, linear functions are not be sufficient, and we might want the mapping functions to be nonlinear. Then, instead of constructing K linear mapping functions, $f_1, \cdots, f_K$, we can directly compute the embedding result of each given instance. In this situation, the "latent" mapping functions can be nonlinear. This problem is technically less challenging, and the corresponding cost function and algorithm can be given in a similar manner as the linear case discussed in this paper.

3 Domain Adaptation

3.1 The Algorithm

Assuming all but one of the input data sets correspond to the source domains, and one input data set corresponds to the target domain, the algorithm to construct $f_1, \cdots, f_K$ by minimizing $\mathcal{C}(f_1, \cdots, f_K)$ is given in Figure 2.

Most existing domain adaptation approaches assume that the input domains are defined by the same features and the

1. **Construct matrices Z, L, L_s and L_d as defined in Section 2.3**.
2. **Compute the mapping functions $(f_1, \cdots, f_K)$ to align the input data sets:**
   $\begin{pmatrix} f_1 \\ \cdots \\ f_K \end{pmatrix} = (\gamma_1, \cdots, \gamma_d)$, where $\gamma_1, \cdots, \gamma_d$ are eigenvectors corresponding to the d smallest eigenvalues of $Z(\mu L + L_s)Z^T x = \lambda Z L_d Z^T x$.
3. **Apply $f_1, \cdots, f_K$ to map the input data sets to the new d dimensional latent space:**
   For any x_a^i and x_b^j, $f_a^T x_a^i$ and $f_b^T x_b^j$ are in the same d-dimensional space and can be directly compared.
4. **Compute solutions for the learning tasks (e.g. classification, ranking) in the latent space with regular learning/transfer learning techniques, leveraging the data from the source domains.**

Figure 2: The Algorithmic Framework.

difference between domains largely comes from data distributions. Our approach projects the input domains defined by different features to a new space, so it can be combined with most existing domain adaptation algorithms to help solve more challenging adaptation problems. Our paper focuses on the construction of common latent space rather than studying which existing domain adaptation approach fits our framework the best. So in the experiments, we compare our algorithm and the other related algorithms on the ability to create such a latent space. A simple domain adaptation approach is applied on top of the latent spaces to see how different algorithms help in heterogeneous domain adaptation.

The domain adaptation approach we use is as follows: In the latent space, we first use the labeled instances from the source domains to train a linear regression model for the given learning task, like ranking/classification. Then we apply manifold regularization [Belkin, Niyogi, and Sindhwani, 2006] as a second linear regression model so that the sum of these two regression scores is close to the desired label for each labeled instance in the target domain. The first regression model makes use of the data from the source domain, while the second regression model adapts the first model to the new domain.

3.2 A Representation Translation Framework Based on Manifold Alignment

Our approach also provides a representation translator across different domains, which can translate any instance from one domain to another. This translator offers us the ability to reuse the existing models rather than just the labels in the source domain. For example, if we already have a ranking model in the source domain, then we can translate any instance from the target domain to the source domain using the translator and then directly apply the existing model to process the translation result. Such a translation is done via the latent space, which only keeps the information that is shared by all the input domains. So the information that is shared across domains can be translated; the information that is only useful for one particular domain will not be translated.

The representation translation framework is illustrated in Figure 3. Without loss of generality, we assume only one source domain X and one target domain Y are given. Following the algorithmic framework, once mapping functions f_X and f_Y are available, we can map instances from each individual data set to the latent space as follows: For any $x_i \in X$, its representation in latent space is $f_X^T x_i$. For any $y_j \in Y$, its representation in latent space is $f_Y^T y_j$. Compared to f_X and f_Y, $f_X f_Y^+$ and $f_Y f_X^+$ go one step further, where '$+$' represents the pseudo inverse. They directly build mappings between input manifolds, and can be used as "translators" to translate instances between spaces. The formulas to translate instances across domains are as follows:

$$\text{For any } x_i \in X, \text{ its representation in } Y \text{ is } (f_X f_Y^+)^T x_i.$$
$$\text{For any } y_j \in Y, \text{ its representation in } X \text{ is } (f_Y f_X^+)^T y_j.$$

4 Applications and Results

To decide μ, we first re-scale W_s and W such that the sum of all elements in W_s equals to the sum of all elements in $W_1, \cdots, W_K$. If matching instances and topology preservations are equally important, $\mu = 1$. If we want to focus more on topology preservation, $\mu > 1$.

4.1 An Illustrative Example

In this example, we directly align the given datasets and use some pictures to illustrate how the alignment algorithms work. The given datasets come from real protein tertiary structure data. A protein 3D structure is a set of amino acids. Let m be the number of amino acids in a given protein and $C_1, \cdots, C_m$ be the coordinate vectors for the amino acids, where $C_i = (C_{i,1}, C_{i,2}, C_{i,3})^T$ and $C_{i,1}, C_{i,2}$, and $C_{i,3}$ are the x, y, z coordinates of amino acid i. In this test, we use two proteins as input data sets and each of which has 215 amino acids. We manually label 5% amino acids in each set as positive, 5% as negative, and the remaining are unlabeled. We denote the first set X_1, the second set X_2, which are both represented by 3×215 matrices. To evaluate how the new algorithm re-scales the input data sets, we manually stretch X_1 by setting $X_1 = 10 \cdot X_1$.

For the purpose of comparison, we plot both datasets on the same graph (Figure 4(A)). It is clear that these two data sets are quite different. The alignment results using the algorithm in Figure 2 are shown in Figure 4(B). In data set 1, a red • represents a positive instance, a blue • represents a negative instance, and a green · represents an unlabeled instance; In data set 2, a red △ represents a positive instance, a blue △ represents a negative instance, and a purple · represents an unlabeled instance. From the results, we can see that both data sets are rescaled to the same size, the positive instances take the left side, and the negative instances take the right side, no matter which domain they are from. In the middle of the figure, some positive and negative instances mix together. The reason for this is that our approach also preserves the topology of the given data set. So the final solution is in

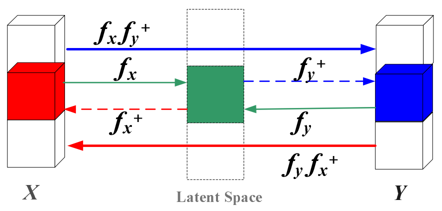

Figure 3: A Representation Translation Framework Based on Manifold Alignment.

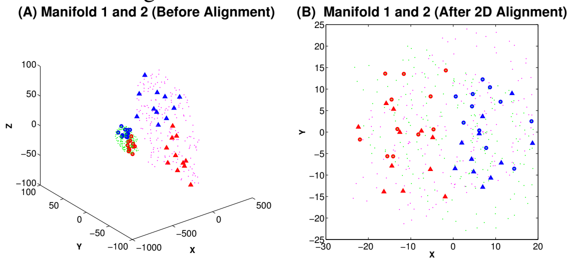

Figure 4: An Illustrative Example.

fact a tradeoff of three goals: matching the instances with the same labels, separating the instances with different labels and preserving the topology for each data set. In this test, $\mu = 1$.

4.2 Text Categorization

The TDT2 corpus consists of data collected during the first half of 1998 and taken from 6 sources. In the data set we are using, the documents that appear in more than one category were removed, and only the largest 30 categories were kept, thus leaving us with 9,394 documents in total. For the purpose of this test, we construct feature sets using two well-known topic modeling algorithms: Latent semantic indexing (LSI) and Latent Dirichlet Allocation (LDA).

We divide the data set into two subsets of the same size, and then learn LSI topics from the first subset and LDA topics from the second subset. We project the first subset onto top 1,000 LSI topics, the second subset onto 1,000 LDA topics. This results in two data sets X_1 and X_2. We assume the labels for all documents in the first subset are available. For the second subset, only 5% documents are labeled. In this test, $\mu = 1$. We first applied our approach to align source and target domains, resulting in a common latent space. Then we applied domain adaptation algorithm on top of this space to learn class categorization for the unlabeled documents in the target domain. For any document x_2^i, the predicted "category label" is a 30×1 vector. We use the probability that the true category is among the top K categories in this label vector to evaluate the performance of different approaches. Note that if we use the largest entry of the label vector to label the document, then the prediction accuracy is equal to the reported result for $K = 1$. For the purpose of comparison, we tested Canonical Correlational Analysis (CCA) and manifold alignment using correspondence under the same setting.We also report the performance of manifold regularization using the data from target domain only. The results are summarized in Figure 5. The new label-based manifold alignment outperformed the other approaches. The performance of CCA was similar to manifold regularization. Manifold regularization does not leverage the data from source domain, while CCA does not take manifold topology into consideration, so it might run into overfitting problems. In this test, we applied all features in the latent space. We also tried using top 1,000 features sorted by eigenvalues, and the performances for two manifold alignment approaches and CCA significantly dropped. Under the new setting, our approach still outperformed all three other approachs (the results are not included in this paper).

4.3 Learning to Rank

In this test, we apply our algorithm to the problem of learning to rank. We assume that for the queries in the training set (source domain), we have a large number of judged documents. For each query in the test set (target domain), we only have judgements for a few documents even though the total number of retrieved documents could be larger than several thousand. The problem is to improve ranking performances for the queries in the test set. We assume that the source and target domains do not share common features.

The data we use in this experiment is the TREC collection used by Aslam et al. [Aslam et al., 2009] to compare the effect of different document selection methodologies for learning to rank. The document corpus, the queries and the relevance judgments in this collection are obtained from TREC 6, 7 and 8 ad-hoc retrieval track. This data set contains 150 queries. The document set we use in our experiments contains the documents from the depth-100 pools of a number of retrieval systems run over these queries. Depth-k pools are constructed by judging only the top k documents from different systems and ignoring the rest of the documents. Each query on average contains approximately 1,500 judged documents, where on average 100 of them are relevant. In this data set, each query-document pair is represented by 22 features. These features are a subset of the LETOR 3.0 features [Liu et al., 2009]. The documents in this data set have two labels: relevant and non-relevant. Relevant documents do not distribute uniformly. Some queries have much less relevant documents than others. For the purpose of test, the data set is split into 5 folds, where each fold contains a training set with 60 queries and a test set with 90 queries. In the training set, all query-document pairs are labeled. In the test set, for each query we have 10 documents that are labeled and roughly 1,500 documents that are not labeled. To simulate a real world problem, we assume 2 features in the source domain are missing. We also apply a rotation and a random noise to the remaining 20 features in the source domain such that the training and test sets do not share common features.

We treat the training set as the source domain; each test query together with its retrieved documents forms the target domain. This results in one source domain and many target domains. We test each target domain independently. Since the source and target domains are defined by different features, most existing domain adaptation approaches will not work for this scenario. Similar to the previous test, we tested our new algorithm, correspondence-based manifold align-

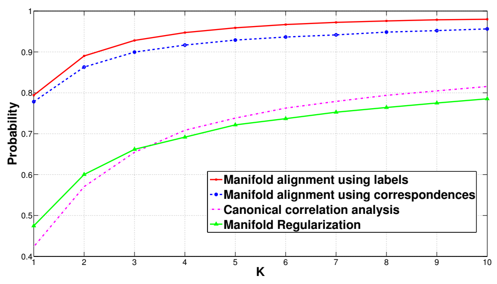

Figure 5: TDT2 Test.

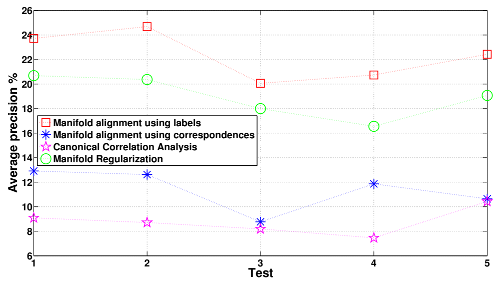

Figure 6: TREC Test.

ment [Wang and Mahadevan, 2009], CCA and manifold regularization (target domain only) using this data. We use the average precision (AP) of each query in the test set to compare the quality of different algorithms. Figure 6 summarizes the average of 90 average precision scores for each fold. The y axis in the plots shows the average precision value and the x axis shows the fold in the data sets used to test the method. In this test, $\mu = 100$ and $d = 40$. Similar to the result of TDT2 test, the new label-based manifold alignment outperformed the other approaches. CCA performed the worst for this task. Interestingly, manifold regularization (target domain only) did a reasonably good job in this test (and also in the previous test). Manifold regularization takes unlabeled data instances in the target domain into consideration. This helps solve overfitting problems even in the case when the labeled information is very limited in the target domain.

5 Conclusions

In this paper, we propose a manifold alignment based approach for heterogeneous domain adaptation. A key aspect of this approach is to construct mappings to link different feature spaces in order to transfer knowledge across domains. The new approach can handle the situation when the input domains do not share any common features or instances. Our paper focuses on the construction of common latent space rather than studying which existing domain adaptation approach fits our framework the best. This paper also extends correspondence based manifold alignment approaches by making use of labels rather than correspondences to align the manifolds. This extension may significantly broaden the application scope of manifold alignment.

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant Nos. NSF CCF-1025120, IIS-0534999, and IIS-0803288.

References

[Aslam et al., 2009] Aslam, J. A.; Kanoulas, E.; Pavlu, V.; Savev, S.; and Yilmaz, E. 2009. Document selection methodologies for efficient and effective learning-to-rank. In *Proceedings of the ACM SIGIR conference on Research and development in information retrieval (SIGIR)*, 468–475.

[Belkin, Niyogi, and Sindhwani, 2006] Belkin, M.; Niyogi, P.; and Sindhwani, V. 2006. Manifold regularization: a geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research* 2399–2434.

[Blitzer, McDonald, and Pereira, 2006] Blitzer, J.; McDonald, R.; and Pereira, F. 2006. Domain adaptation with structural correspondence learning. In *Proceedings of the Conference on Empirical Methods on Natural Language Processing*, 120–128.

[Dai et al., 2008] Dai, W.; Chen, Y.; Xue, G.; Yang, Q.; and Yu, Y. 2008. Translated learning: transfer learning across different feature spaces. In *Proceedings of the Advances in Neural Information Processing Systems (NIPS)*, 353–360.

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

[Duan et al., 2009] Duan, L.; Tsang, I.; Xu, D.; and Chua, T. 2009. Domain adaptation from multiple sources via auxiliary classifiers. In *Proceedings of the International Conference on Machine Learning (ICML)*, 103–112.

[Ham, Lee, and Saul, 2005] Ham, J.; Lee, D.; and Saul, L. 2005. Semisupervised alignment of manifolds. In *Procedings of the International Workshop on Artificial Intelligence and Statistics*, 120–127.

[Liu et al., 2009] Liu, T.; Qin, T.; Xu, J.; Xiong, W.; and Li, H. 2009. Letor: benchmark dataset for research on learning to rank for information retrieval, http://research.microsoft.com/users/letor/.

[Mansour, Mohri, and Rostamizadeh, 2009] Mansour, Y.; Mohri, M.; and Rostamizadeh, A. 2009. Domain adaptation with multiple sources. In *Proceedings of the Advances in Neural Information Processing Systems (NIPS)*, 1041–1048.

[Pan and Yang, 2010] Pan, S. J., and Yang, Q. 2010. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering* 22(10):1345–1359.

[Wang and Mahadevan, 2009] Wang, C., and Mahadevan, S. 2009. A general framework for manifold alignment. In *AAAI Fall Symposium on Manifold Learning and its Applications*.

[Wilks, 1963] Wilks, S. S. 1963. *Mathematical statistics*. Wiley.