Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)

Cross-Domain Kernel Induction for Transfer Learning

Wei-Cheng Chang*
Carnegie Mellon University
wchang2@andrew.cmu.edu

Yuexin Wu*
Carnegie Mellon University
yuexinw@andrew.cmu.edu

Hanxiao Liu
Carnegie Mellon University
hanxiaol@cs.cmu.edu

Yiming Yang
Carnegie Mellon University
yiming@cs.cmu.edu

Abstract

The key question in transfer learning (TL) research is how to make model induction transferable across different domains. Common methods so far require source and target domains to have a shared/homogeneous feature space, or the projection of features from heterogeneous domains onto a shared space. This paper proposes a novel framework, which does not require a shared feature space but instead uses a parallel corpus to calibrate domain-specific kernels into a unified kernel, to leverage graph-based label propagation in cross-domain settings, and to optimize semi-supervised learning based on labeled and unlabeled data in both source and target domains. Our experiments on benchmark datasets show advantageous performance of the proposed method over that of other state-of-the-art TL methods.

1 Introduction

Transfer learning (TL) aims to address the label-sparse problem arising in many real-world applications as acquiring a large quantity of labeled data is extremely expensive and labor-intensive. TL methods address this problem by transferring the trained models from label-rich domain (source domain) to a relevant but label-sparse domain (target domain) according for the task of interest. Using topic classification of web blogs as an example (as in (Pan et al. 2011)), obtaining a large set of labeled instances is often difficult especially when the web blogs are newly released. On the other hand, large collections of labeled news stories in relevant topics may be easily found on the internet. Thus if we can successfully transfer the classification models or the induced features from the news-story domain to the web blog domain, then the label-sparse problem in the target domain would be effectively addressed. Another motivating example is to transfer the text classification models from a label-rich language (e.g., English) to a label-sparse or label-sparser language (e.g., Italian or Turkish). Unlike English or a few internationally dominating languages, most of the other languages in the world have much less labeled documents in comparison. This means that TL would have a tremendous impact on the true success of text classification for all languages in the world if we can solve TL in all the cross-lingual settings. Notice an important difference between the two examples we have introduced, i.e., in the first example both source and target domain share the same feature space (the same vocabulary of English), while in the second example the two domains have different feature spaces (i.e., the vocabularies of two different languages). Nevertheless, TL across different feature spaces (*heterogeneous*) is usually a tougher problem than TL within the common feature space (*homogeneous*).

The literature of TL methods (Pan and Yang 2010) reveals promising results in a variety of real-world applications, such as text classification (Pan et al. 2011; Duan, Xu, and Tsang 2012), image classification (Zhu et al. 2011; Kulis, Saenko, and Darrell 2011), sentiment analysis (Glorot, Bordes, and Bengio 2011; Zhou et al. 2014), recommendation systems (Li, Yang, and Xue 2009), and more. Let us outline the major differences among existing approaches based on their basic assumptions in relating source and target domain, as well as on how labeled and unlabeled data in both domains are jointly leveraged during the TL process.

The Naive Bayes Transfer Classifier (NBTC) (Pan et al. 2011) is a representative work on TL for text classification, under the assumption that source and target domain share the same feature space as well as the topic labels. However, the topic distribution and the distributions of topics conditioned on words may differ in two domains. The goal of NBTC is to adapt source-domain distributions to the target-domain distributions. A major limitation of the NBTC approach, and any other methods under the same assumption of a shared feature space between the two domains, is that they cannot handle TL across heterogeneous feature spaces. For example, those methods are not applicable for performing TL from text classification to image classification (and vice versa), or from classification of English documents to that of other languages.

Yet another kind of approaches, *Transductive Transfer Learning* (TTL), tackles TL problem from a different angle. TTL focuses on cross-domain kernel construction and the utilization of unlabeled data in *both* source and target domains during the learning procedure. Adaptation Regularization based Transfer Learning (ARTL)(Long et al. 2014) is a representative work of TTL methods. It constructs a unified kernel and applies graph-based label propagation technique under certain regularized constraints to infer labels

*Both student authors contributed equally.

in target domain. This kernel-based approach is highly effective even given limited training data. However, ARTL or any existing TTL-based methods, to the best of our knowledge, is not applicable to the heterogeneous feature setting, which is the focus of this paper. The difficulty arises in cross-domain kernel construction. How could a kernel value between data from different domains be computed if they are not in the same feature space?

One common stream of approaches, *Feature Representation Transfer Learning* (FRTL), which can be used in heterogeneous settings addresses the problems by learning a common feature space, and then performing model transfer or parameter adaptation within that subspace. An important assumption adopted by most existing FRTL approaches is the availability of cross-domain *parallel data*. i.e., corresponding instances that have both source and target representations. There are various way to learn the common feature representation. For example, (Argyriou et al. 2007) tried to induce a shared projection matrix for both source and target domain. (Glorot, Bordes, and Bengio 2011) applied a deep learning technique, the stack denoised autoencoder (SDA), for a non-linear projection onto the shared latent space in cross-domain sentiment classification. (Chandar et al. 2015) proposed a correlational neural network (CorrNet) approach that combines autoencoders (AE) and canonical correlation analysis (CCA) in the way that AE learns a generalized representation for each domain while CCA captures the joint representation of the two domains by maximizing the correlation in-between. Notice that above state-of-the-art neural network methods (CorrNet) usually require large amount of parallel data to achieve competitive results. Such large size of parallel data may not be realistic to obtain given fixed budgeted resources in real world applications. (e.g. human labeling for parallel sentences in low-resource languages)

In this paper, we propose a novel framework called *Kernel Induction for Heterogeneous Feature TL* (KerTL). Our approach addresses the limitation of TTL methods by introducing a powerful kernel completion technique. This enables our approach not only to enjoy same degree of smooth label propagation as in TTL from source to target domain but also to require only a modest amount of parallel data as opposed to neural-network based methods. Furthermore, using low-rank spectral transformation of the component kernels to obtain the global approximation of kernel diffusion leads to additional power and computational efficiency of our framework.

2 Proposed Framework

Let us first formally define the Transfer Learning (TL) problem of interest, and then show how to formulate TL as an optimization problem with graph regularization.

TL Definitions

For any single data domain $\mathcal{D} = \mathcal{X} \times \mathcal{Y}$, denote by $D = O \cup U$ the training set consisting of both labeled examples $O = \{\boldsymbol{x}_i, y_i\}$ and unlabeled examples $U = \{\boldsymbol{x}_i\}$ drawn from $\mathcal{X} \times \mathcal{Y}$ and $\mathcal{X}$ respectively. If the context is clear, we abuse the notation of $\boldsymbol{x}$ to denote a feature vector in D without distinguishing whether it comes from O or U.

In this paper, we focus on TL involving two domains with *heterogeneous* features but a shared label space. Specifically, we are given a source domain $\mathcal{D}_s = \mathcal{X}_s \times \mathcal{Y}$ and a target domain $\mathcal{D}_t = \mathcal{X}_t \times \mathcal{Y}$ where $\mathcal{X}_s$ is allowed to differ from $\mathcal{X}_t$. We in addition assume the accessibility to a parallel set $PL = \{(\boldsymbol{x}_i^{(pls)}, \boldsymbol{x}_i^{(plt)})\}$ where $\boldsymbol{x}_i^{(pls)} \in D_s$, $\boldsymbol{x}_i^{(plt)} \in D_t$. Each feature pair in PL corresponds to "one" datum's representation in two domains. Given D_s, D_t and PL, our goal is to make predictions on the unlabeled target-domain data U_t with low expected error.

Notice that all data points are already specified in $D_s \cup D_t$. The parallel set PL only suggests inter-domain relations, namely which data in D_s have counter-parts in D_t.

TL with the Graph Laplacian

The aforementioned parallel-data-based TL problem can be formulated in a way that is compatible with graph-based SSL. Specifically, we view all (both labeled and unlabeled) data points in $D_s \cup D_t$ as nodes in a graph, whose edges encode inter-node similarities summarized in a $|D_s \cup D_t| \times |D_s \cup D_t|$ adjacency matrix W (Section 3). Our task therefore becomes making predictions on the target-domain unlabeled nodes (U_t) in the graph. With limited supervision available, it is desirable to propagate from labeled nodes to unlabeled ones with respect to the manifold structure of the graph, on the assumption that nodes sharing high similarities should also share similar labels.

For brevity we assume a binary label space $\mathcal{Y} = \{-1, 1\}$. Denote by y the true label and by $f(\boldsymbol{x})$ the corresponding predicted value. Predicted values over all nodes in $D_s \cup D_t$ are further concatenated to form a long vector $\boldsymbol{f} = [\boldsymbol{f}_s, \boldsymbol{f}_t]^\top$ of length $|D_s \cup D_t|$. Our problem is then formalized as:

$$\min_{\boldsymbol{f}} \sum_{(\boldsymbol{x},y) \in O_s \cup O_t} \ell(f(\boldsymbol{x}), y) + \gamma \boldsymbol{f}^\top \mathcal{L} \boldsymbol{f}, \qquad (1)$$

where $\mathcal{L}$ is the graph Laplacian characterizing smoothness of graph and γ is a positive scalar controlling the regularization strength. Laplacian $\mathcal{L}$ is defined as $\mathcal{L} = W\mathbf{1} - W$.

The term in (1) is the empirical loss between predicted labels and true labels on subsets O_s and O_t. While the last term indicates the normalization penalty with respect to the manifold structure of all data. Specifically, we can show that this Laplacian can be reformulated as $\boldsymbol{f}^\top \mathcal{L} \boldsymbol{f} = \frac{1}{2} \sum_{i,j} w_{ij} (f_i - f_j)^2$, which reveals the motivation of the penalty: nodes that have strong similarities (with large w_{ij}) should have close prediction scores (f_i and f_j).

3 Graph Construction

The adjacency matrix W and its associated Laplacian $\mathcal{L}$ play a key role in our formulation. In the homogeneous setting, there are widely-accepted routines to compute W using either cosine or radial basis function (RBF) measurements. Nonetheless, under the heterogeneity assumption, all those methods would become inappropriate in evaluating similarities for the inter-domain part. This implies the need of completing (instead of directly computing) the inter-domain similarities through both the pre-computed intra-domain similarities and the information from the parallel data, which

is one of the key contributions in our work. In the following, we start from homogeneous graph construction and then move on to the more generic framework of tackling heterogeneous scenarios.

Homogeneous Graph Construction

Given a pair of data points $\boldsymbol{x}_i, \boldsymbol{x}_j$ in homogeneous feature space $\mathcal{X}$, one could compute the pair-wise similarity w_{ij} using different functions, among which two typical choices are cosine measurement and RBF measurement $w_{ij} = \exp(-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{\sigma^2})$. The choice of similarity function is usually domain-specific. For example, with text data (term frequency), cosine measurement is empirically often a better choice in characterizing documents with similar (proportional) word counts.

Besides, one would usually consider "dropping out" some weights (i.e. truncating a subset of w_{ij}'s to 0) which is called "sparsification" in order to emphasize local information and to lower the computation cost. A common practice is to keep weights only of each node's k nearest neighbors (kNN). Compared to ϵ-graphs where one specifies a fixed threshold for truncating all edge weights, the kNN graph allows "adaptive" neighborhood radius for both strongly and weakly connected points (Zhu, Lafferty, and Rosenfeld 2005), and often leads to better classification results.

Heterogeneous Graph Construction

The construction for intra-domain similarities stays valid within the heterogeneity setting. However, the inter-domain scores cannot be directly computed (neither $\langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle$ (cosine) nor $\|\boldsymbol{x}_i - \boldsymbol{x}_j\|$ (RBF) can be calculated as $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are in different feature spaces). To simplify our discussion, suppose the data are in a well-arranged order such that the adjacency matrix W is in the following form:

$$W = \begin{bmatrix} W_{s,s} & W_{s,t} \\ W_{t,s} & W_{t,t} \end{bmatrix}, \tag{2}$$

where $W_{s,s}$, $W_{t,t}$ represent the intra-domain parts, and $W_{s,t} = W_{t,s}^\top$ represent the inter-domain parts.

Suppose $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are a pair of parallel data, which means they are two alternative views of one datum. Then, it is always reasonable to set $w_{ij} = 1$ since a datum and itself should always be similar to itself. However, these entries are the only observable cells in $W_{s,t}$ and $W_{t,s}$. All the remaining cells should be completed using information from intra-domain similarities and parallel data.

Such off-diagonal matrix completion problem has strong resemblance with the bipartite graph edge completion problem, where nodes in D_s and D_t form a bipartite graph and the goal is to complete the bipartite edges ($W_{s,t}$) in the middle (Liu and Yang 2015). In the following, we use $\hat{W}_{s,t}$ to denote the completed matrix and $W_{s,t}$ for the original (observed) version.

Random Walk Completion Let us consider the task of completing the missing (p, q)-th entry in $W_{s,t}$. Although the value of $(w_{s,t})_{pq}$ is unknown, some other entry (r, q) in the same column might have been observed and hence $(w_{s,t})_{pq}$ should be close to $(w_{s,t})_{rq}$ if the two "must-links" (p, q) and (r, q) are similar. Such similarity is provided as $(w_{s,s})_{pr}$. This suggests completing $(w_{s,t})_{pq}$ by aggregating all elements in the q-th column of $W_{s,t}$ weighted by the p-th row in $W_{s,s}$. Namely $(\hat{w}_{s,t})_{pq} \leftarrow \sum_r (w_{s,s})_{pr}(w_{s,t})_{rq}$. The above can be expressed in the matrix form

$$\hat{W}_{s,t} \leftarrow W_{s,s} W_{s,t}. \tag{3}$$

When $W_{s,s}$ is normalized as a column-stochastic matrix, equation (3) amounts to one-step random walk for each column in $W_{s,t}$.

Alternatively, completion can be carried out row-wisely

$$\hat{W}_{s,t} \leftarrow W_{s,t} W_{t,t}. \tag{4}$$

Combining (3) and (4) leads to one-step simultaneous random walk in both the source and target domain:

$$\hat{W}_{s,t} \leftarrow W_{s,s} W_{s,t} W_{t,t}. \tag{5}$$

By further allowing varying number of random walk steps on both sides (k steps on both sides in total), and by aggregating the effect of all different steps, we obtain

$$\hat{W}_{s,t}^{(k)} = \sum_{i=0}^{k} \binom{k}{i} W_{s,s}^i W_{s,t} W_{t,t}^{k-i}. \tag{6}$$

Compared to one-step random walk completion, (6) takes into account multi-step transduction over the graph, which is particularly desirable in our case where missing entries in $W_{s,t}$ may not have observed entries as its direct neighbor.

t

Diffusion Kernel Completion We propose the following diffusion kernel completion

$$\hat{W}_{s,t} = \exp(\alpha_s W_{s,s}) W_{s,t} \exp(\alpha_t W_{t,t}). \tag{7}$$

This is equivalent to the aggregation of infinite number of weighted Random Walk Completions. Specifically,

$$\hat{W}_{s,t} = \sum_{k=0}^{\infty} \hat{W}_{s,t}^{(k)}(\alpha_s, \alpha_t). \tag{8}$$

where $\hat{W}_{s,t}^{(k)}$ denotes the weighted Random Walk Completion:

$$\hat{W}_{s,t}^{(k)}(\alpha_s, \alpha_t) = \sum_{i=0}^{k} \binom{k}{i} \alpha_s^i W_{s,s}^i W_{s,t} \alpha_t^{k-i} W_{t,t}^{k-i}. \tag{9}$$

positive scalars α_s and α_t are corresponding to the weights for the source- and target- domain graphs, respectively. Due to space limit, we do not provide the proof details.

Low Rank Approximation of Diffusion Kernel As in many other matrix completion tasks, it can be useful to impose low-rank assumptions on $\hat{W}_{s,t}$. The compressed sensing theory (Candès and Recht 2009) implies there is still hope to recover $\hat{W}_{s,t}$ even if our intra-domain matrices are non-informative (e.g. identity matrices). To some extent, the low-rank factorization process is a denoising procedure trying to recover the missing signals.

Therefore, we first take the low-rank eigen-decomposition on both $\exp(\alpha_s W_{s,s})$ and $\exp(\alpha_t W_{t,t})$ such that

$$\exp(\alpha_s W_{s,s}) \approx Q_s \exp(\alpha_s \Lambda_s) Q_s^\top \tag{10}$$

$$\exp(\alpha_t W_{t,t}) \approx Q_t \exp(\alpha_t \Lambda_t) Q_t^\top, \tag{11}$$

where Λ_s, Λ_t are the k_s, k_t leading eigen-values for $W_{s,s}$, $W_{t,t}$ respectively, and Q_s, Q_t are the corresponding stacked eigen-vectors for $W_{s,s}$, $W_{t,t}$.

The diffusion kernel completion (7) is then modified as

$$\hat{W}_{s,t} = \left(Q_s \exp(\alpha_s \Lambda_s) Q_s^\top\right) W_{s,t} \left(Q_t \exp(\alpha_t \Lambda_t) Q_t^\top\right). \tag{12}$$

4 Optimization Algorithms

The proposed graph construction method gives us a joint adjacency matrix W for all data points in both the source and target domains, along with its associated graph Laplacian. To recap, our task is to solve the optimization problem:

$$\min_{\boldsymbol{f}} \quad h(\boldsymbol{f}) \equiv \sum_{i \in \mathcal{O}_f \cup \mathcal{O}_\sqcup} \ell(f_i, y_i) + \gamma \boldsymbol{f}^\top \mathcal{L} \boldsymbol{f}, \tag{13}$$

It is not hard to verify that (13) is a convex optimization problem when $\ell(\cdot, \cdot)$ is convex. This enables us to adopt a wide range of optimization techniques. In particular, we compute the exact solution for the square loss and use Adagrad which is a widely-tested sub-gradient method (Duchi, Hazan, and Singer 2011) for other losses (e.g. logistic and hinge loss).

Note that our computation could be fast when using the low-rank approximation. The computational bottleneck of our method during optimization lies in the multiplication of $\mathcal{L}$ and $\boldsymbol{f}$ when calculating the gradient of (13). Recall $\mathcal{L}$ is a function of W, the gradient computation can be carried out in linear time over $|\mathcal{D}_t|$ and $|\mathcal{D}_s|$ when the diagonal blocks $W_{s,s}$, $W_{t,t}$ take kNN forms, making their multiplication with $\boldsymbol{f}$ cost as much as $O(k|\mathcal{D}_s|)$ and $O(k|\mathcal{D}_s|)$, respectively. Similarly, the time complexity of doing matrix-vector multiplication with the off-diagonal block $\hat{W}_{s,t}$ will be $O(d \min(|\mathcal{D}_s|, |\mathcal{D}_t|))$ where d is the low-rank dimension.

5 Experiments

Datasets

Amazon Product Reviews (APR) The APR dataset (Prettenhofer and Stein 2010) was designed for evaluations of sentimental classification with transfer learning in cross-language and cross-domain settings. It consists of Amazon product reviews on books (B), DVDs (D) and music (M), and written in English (EN), German (GE), French (FR) and Japanese (JP). For each language on each product type (B, D or M), there are 2000 labeled reviews for training and 2000 labeled reviews for testing, respectively. Parallel data are also provided for each language pair, which we will describe with an example *task* in the next.

Following the settings in (Zhou et al. 2014), we treat English as the source language, and the remaining three languages (German, French and Japanese) as the target languages. For each language pair (EN-GE, EN-FR or EN-JP), we have 6 cross-product-type pairs, constituting overall 18 cross-language and cross-product-type combinations (e.g. EN-B-FR-D as shown in the first column of Table 2). Specifically, EN-B-FR-D represents TL task with English reviews on Books as source domain, and French reviews on DVD products as target domain. The parallel dataset for the EN-B-FR-D task is obtained by running Google translation over the 2,000 French book reviews in the training set, and by treating the system-produced translations as the English behalf of the parallel data.

MNIST Handwritten Images The MNIST dataset consists of $70,000$ images in total, with digits from 0 to 9 as the class labels (one per image). We follow the setting in (Chandar et al. 2015), to treat *left* half of each image (28×28 pixels) as a source-domain instance, while *right* half of the image as a target-domain instance. Raw pixel values are used as features. We randomly sampled $3,000$ images from the full set as the unlabeled parallel set, $2,000$ images as the source-domain training set, $1,024$ images as the target-domain training set, and another of $2,000$ images as the test set (only the target-domain portion is used). We call the classification with respect to each target label (a digit from 0 to 9) as a task in image recognition. Although the source and target domains have same feature dimensions, the features are indeed heterogeneous (direct cosine/RBF computation of two half images would not indicate label similarity). The idea would be more clear if we cut images in a 1/3 and 2/3 fashion, but for the ease of comparison with existing methods, we keep the same cutting scheme (Chandar et al. 2015).

Constructing unbalance training sets and size-varying parallel sets To simulate the label-sparse condition of target domain as in TL problem, we construct unbalanced training set for our experiments on the APR and MNIST data sets. Recall that in TL each training set has the source-domain part and the target-domain part, respectively. For each task in APR, we use the full set of 2000 source domain labeled instances, and a randomly sampled subset of m target domain labeled instances ($m = 2, 4, 8, 16, 32$) from the full set as the final training data. The remaining target-domain labeled instances ($2000 - m$) are used for validation (hyper-parameter tuning). In the MNIST data set, the source domain training pool has the full size of $2,000$ instances. Another m instances (for $m = 2, 4, 8, 16, 32$) randomly sampled from the target-domain training set are used to complete the full training set.

As for the parallel data set in each task, we also randomly sampled from the available pool with the sample sizes of $l = 64, 128, 256, 512, 1024, 2000$ for APR ($l = 64, 128, 256, 512, 1024, 2048, 3000$ for MNIST). The size-reduced samples allow us to evaluate transfer learning under the label-unbalanced and parallel-data-sparse conditions. For each value of m and l, we repeated the random sampling 10 times, and averaged the performance of the target-domain classifiers over the randomly sampled training sets and parallel data for each task in the evaluation.

Table 1 summarizes the statistics of the datasets we used in our experiments.

Data sets	APR	MNIST
Source domain training set	2000	2000
Target domain training set	2000	1024
Target domain test set	2000	2000
Parallel data size	2000	3000

Table 1: Data Statistics

Methods for Comparison

We include six methods as baselines for comparison. Two of them are representative methods (SVM and SSL described below) in supervised classification where only the target-domain labeled data are used for training classifiers. We also include two state-of-the-art methods (HFA and MMDT) in transfer learning, which can use labeled data in both the source domain and the target domain for training but cannot leverage parallel data. The remaining two methods (HHTL and CorrNet) are the state-of-the-art TL methods which can use both the labeled data in both domains as well as parallel data in addition. Some details of these baseline methods are described below.

- Support Vector Machine (SVM): We used the L2-SVM from LIBLINEAR (Fan et al. 2008). [1]
- Semi-Supervised Learning (SSL) (Zhu, Lafferty, and Rosenfeld 2005): We implemented the graph-based label propagation method for Semi-Supervised Learning framework.
- Heterogeneous Feature Augmentation (HFA) (Li et al. 2014): This method embeds heterogeneous domain data into shared high-dimensional space, and deploys a Multiple Kernel Learning solver (Kloft et al. 2011). We used the code from the website [2].
- Max Margin Domain Transform (MMDT) (Hoffman et al. 2013): This method uses an asymmetric transformation matrix to map features across domains, which is optimized with respect to all the target categories. We used the code from the website [3].
- Hybrid Heterogeneous Transfer Learning (HHTL) (Zhou et al. 2014): This method uses a parallel corpus to learn the hidden layers which are shared by both the source domain and the target domain, and allow classifiers to be trained on the labeled data in both domains after projecting them onto the shared hidden layer. We used the code provided by the authors.
- Correlational neural network (CorrNet) (Chandar et al. 2015): This method uses autoencoders to simultaneously minimize classification errors in both domains, and to capture cross-domain correlations based on a parallel dataset. Similar to HHTL, classifiers are trained after the data are mapped onto the shared latent space. We used the code from the website [4].

[1] https://www.csie.ntu.edu.tw/∼ cjlin/liblinear/

[2] https://github.com/transmatrix-github/HFA_release

[3] https://github.com/jhoffman/MaxMarginDomainTransforms

[4] https://github.com/apsarath/CorrNet

Detailed Experimental Settings

Our experimental results involve two random factors. The first comes from the random sampling of the target domain labeled training sets, and the second comes from the random sampling of the parallel datasets, as we described in Section 5. All the experiments with random samples are repeated 10 times with different random seeds. Mean and standard deviation of the Area under Curve (AUC) of ROC are reported for evaluation and comparison.

In HHTL and CorrNet, after learning the projected matrices, we trained linear SVMs (Fan et al. 2008) on the projected training data. For all the methods using SVM classifiers (in SVM, HFA, MMDT, HHTL and CorrNet), we set the regularization parameter $C = 1$.

For hyper-parameter tuning, we set the default hyper-parameters of HFA and MMDT the same as in their papers. We adopted the hyper-parameter of HHTL on the APR data, with a grid search of the optimal regularization coefficient among $\lambda = 0.001, 0.01, 1, 10$, and 100, and the corruption probability among $p = 0.5, 0.6, 0.7, 0.8$, and 0.9 on the MNIST dataset. Similarly, for CorrNet on the MNIST dataset we used a grid search for the number of hidden units as $20, 50, 100$, and 200, and $\lambda = 0.2, 2$, and 20 on the APR dataset. For KerTL, we used the cosine similarity and RBF kernel on the APR and MNIST datasets, respectively. We keep the top 128 eigenvectors in the eigen-decomposition part for efficient computation, and set the regularization coefficient γ to be 2^{-10}.

Results

TL methods vs. non-TL methods In the first set of experiments we fixed the training-set size in the target domain as $m = 2$, and the parallel-set size as $l = 1024$. Figure 1 shows the averaged AUC scores of those methods. All the TL methods which leverage parallel data (HHTL, CorrNet and KerTL) significantly outperformed the methods that cannot take advantage of parallel data (SVM, SSL, HFA and MMDT). Among the TL methods, our KerTL outperforms all the other methods on both the APR and MNIST data sets. Tables 2 and 3 show the task-specific performance scores on the two data sets, respectively. Again, the performance of KerTL dominates across most of those tasks. On the MNIST dataset (Table 3) in particular, KerTL improved the result of CorrNet (which is the strongest baseline) from 93.2% to 96.2% in AUC, which is equivalent to reducing the error rate from 6.8% to 3.8%, i.e., a 44.1% reduction in error. Such an improvement is indeed significant.

TL methods with varying-sized parallel data The second set of experiments compares the performance of TL methods (HHTL, CorrNet and KerTL) with varying sized parallel data, while the training-set size is fixed as $m = 2$ in the target domain. As shown in Figure 2, KerTL outperforms HHTL and CorrNet in most regions of the parallel-set sizes, on both the ARP and NMIST data sets. We also observed that the performance of HHTL was very sensitive to the settings of its hyper-parameters. When we fixed those parameter values and varied the sizes of the parallel data, HHTL's

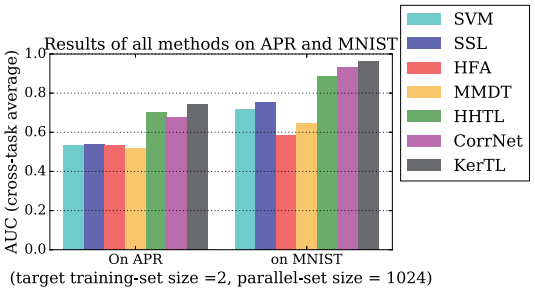

Figure 1: Comparison of all methods on APR and MNIST.

Tasks	SVM	SSL	HFA	MMDT	HHTL	CorrNet	KerTL
EN-B-GE-D	0.564	0.558	0.550	0.563	0.707	0.604	**0.715**
EN-B-GE-M	0.500	0.542	0.536	0.528	0.711	0.659	**0.730**
EN-B-FR-D	0.525	0.513	0.522	0.513	0.747	0.729	**0.748**
EN-B-FR-M	0.541	0.540	0.544	0.542	0.687	0.717	**0.738**
EN-B-JP-D	0.528	0.527	0.541	0.524	0.643	0.692	**0.713**
EN-B-JP-M	0.534	0.537	0.541	0.505	0.611	0.665	**0.724**
EN-D-GE-B	0.502	0.509	0.499	0.482	0.772	0.692	**0.796**
EN-D-GE-M	0.500	0.542	0.517	0.531	0.737	0.672	**0.755**
EN-D-FR-B	0.548	0.549	0.547	0.514	0.743	0.739	**0.785**
EN-D-FR-M	0.541	0.540	0.537	0.543	0.724	0.696	**0.741**
EN-D-JP-B	0.549	0.561	0.558	0.516	0.694	**0.719**	0.717
EN-D-JP-M	0.534	0.537	0.536	0.506	0.683	**0.747**	0.713
EN-M-GE-B	0.502	0.509	0.520	0.476	0.704	0.668	**0.786**
EN-M-GE-D	0.564	0.558	0.519	0.561	0.728	0.631	**0.740**
EN-M-FR-B	0.548	0.549	0.542	0.515	0.745	0.672	**0.789**
EN-M-FR-D	0.525	0.513	0.508	0.511	0.755	0.670	**0.764**
EN-M-JP-B	0.549	0.561	0.526	0.529	0.622	0.675	**0.739**
EN-M-JP-D	0.528	0.527	0.551	0.487	0.655	0.707	**0.708**
Average ± Std	0.532 ±0.021	0.537 ±0.018	0.533 ±0.016	0.519 ±0.023	0.704 ±0.047	0.687 ±0.037	**0.745** ±**0.029**

Table 2: Overall results on APR dataset with target domain training-set size of 2 and parallel set size of 1024. Bold-faced numbers indicate the best result on each row.

performance was either unstable (on MNIST) or decreasing (on APR) as the parallel data size increased.

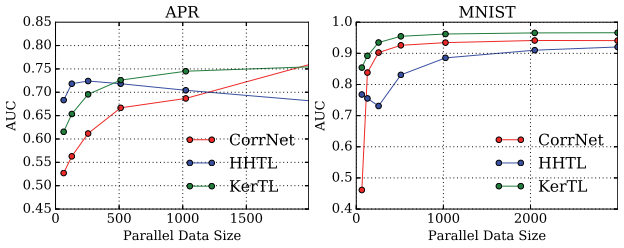

Figure 2: CorrNet, HHTL and KerTL on the APR dataset (left) and the MNIST dataset (right) with a varying quantity of parallel data.

Influence of label sparsity in the target domain The third set of experiments compares KerTL with SVM, SSL, HFA and MMDT under the condition that the labeled training instances are extremely sparse in the target domain, specifically with $m = 2, 4, 8, 16, 32$. Size of parallel datasets are $l = 1024$ in those experiments.

Figure 3 shows results on APR and MNIST datasets. On both data sets, the curves of SVM, SSL and HFA increase rapidly when the training-set sizes are below 200. Without

Tasks	SVM	SSL	HFA	MMDT	HHTL	CorrNet	KerTL
Digit 0	0.950	0.965	0.891	0.855	0.971	0.987	**0.989**
Digit 1	0.906	0.915	0.500	0.838	0.989	0.994	**0.996**
Digit 2	0.699	0.739	0.539	0.567	0.867	0.931	**0.962**
Digit 3	0.628	0.785	0.637	0.664	0.861	0.892	**0.939**
Digit 4	0.672	0.613	0.500	0.477	0.867	0.937	**0.958**
Digit 5	0.598	0.607	0.500	0.543	0.774	0.877	**0.959**
Digit 6	0.848	0.877	0.677	0.536	0.937	0.962	**0.985**
Digit 7	0.714	0.736	0.441	0.686	0.919	0.956	**0.968**
Digit 8	0.494	0.592	0.720	0.651	0.823	0.890	**0.936**
Digit 9	0.674	0.690	0.430	0.623	0.846	0.918	**0.929**
Average ± Std	0.718 ±0.143	0.752 ±0.133	0.584 ±0.138	0.644 ±0.118	0.885 ±0.067	0.934 ±0.041	**0.962** ±**0.023**

Table 3: Overall results on MNIST dataset with target domain training-set size of 2 and parallel set size of 1024. Bold-faced numbers indicate the best result on each row.

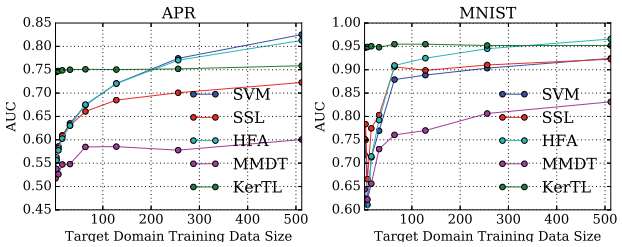

Figure 3: SVM, SSL and KerTL on the APR dataset (left) and the MNIST dataset (right) with a varying quantity of labeled data in the target domain.

leveraging parallel data, MMDT does not perform well on both data set as target domain training data increase. We suspect the reason is that when source and target domain is very different (APR and MNIST in our setting), linear transform of the mapping with max margin criterion is not possible to find good representation without the help of parallel data. On the other hand, HFA is only comparable to KerTL when target domain training data is large enough since it does not utilize parallel data. KerTL has a nearly flat curve, substantially outperforming the others in the label-sparse regions. This implies KerTL could successfully transferred source-domain training data especially when source and target domain ($m = 2 \sim 32$) training data are very imbalanced.

But why the performance curve of KerTL is below that of SVM when the training-set size in the target domain is beyond 200 on the APR data? We believe that it is caused by the *imperfect* parallel data we used in KerTL. Recall that in our previous example of the EN-B-FR-D task, the parallel data are the paired English/French book reviews, assuming that the ideal parallel set of (manually aligned) English book reviews and French DVD reviews are not available. In other words, the parallel data provided in APR has a domain mismatch with respect to the reviews on different product types, which is most helpful when the label-sparse issue is severe.

In contrast, the parallel data sets in MNIST do not have a domain mismatch issue, as each pair in the parallel set consists of the left-half (as the source-domain instance) and right-half (as the target-domain instance) in the same image. We argue that the APR way of constructing parallel data is

more realistic than that in MNIST, because we usually cannot get each image instance halfly labeled and halfly unlabeled in real-word applications of image classification.

6 Conclusions

In this paper we proposed a novel framework for transfer learning with cross-domain kernel induction. Our approach uses a parallel corpus to calibrate domain-specific graph Laplacians into a unified kernel, and to optimize semi-supervised label propagation based on the labeled and unlabeled data in both domains. Our extensive experiments show that all the TL methods in our evaluation significantly outperformed non-TL ones (SVM and SSL), and that the proposed method outperforms other state-of-the-art TL methods (HFA, MMDT, HHTL and CorrNet) when the target-domain labeled data are extremely sparse and the quantity of available parallel data is also limited. Those results indicates cross-language and cross-domain kernel induction is a promising direction to pursue in transfer learning.

7 Acknowledgment

We thank the reviewers for their helpful comments. This work is supported in part by the National Science Foundation (NSF) under grant IIS-1546329.

References

Argyriou, A.; Pontil, M.; Ying, Y.; and Micchelli, C. A. 2007. A spectral regularization framework for multi-task structure learning. In *Advances in neural information processing systems*, 25–32.

Candès, E. J., and Recht, B. 2009. Exact matrix completion via convex optimization. *Foundations of Computational mathematics* 9(6):717–772.

Chandar, S.; Khapra, M. M.; Larochelle, H.; and Ravindran, B. 2015. Correlational neural networks. *Neural computation*.

Duan, L.; Xu, D.; and Tsang, I. W. 2012. Learning with augmented features for heterogeneous domain adaptation. In *Proceedings of the International Conference on Machine Learning*, 711–718. Edinburgh, Scotland: Omnipress.

Duchi, J.; Hazan, E.; and Singer, Y. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *The Journal of Machine Learning Research* 12:2121–2159.

Fan, R.-E.; Chang, K.-W.; Hsieh, C.-J.; Wang, X.-R.; and Lin, h. C.-J. 2008. Liblinear: A library for large linear classification. *The Journal of Machine Learning Research* 9:1871–1874.

Glorot, X.; Bordes, A.; and Bengio, Y. 2011. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, 513–520.

Hoffman, J.; Rodner, E.; Donahue, J.; Darrell, T.; and Saenko, K. 2013. Efficient learning of domain-invariant image representations. *arXiv preprint arXiv:1301.3224*.

Kloft, M.; Brefeld, U.; Sonnenburg, S.; and Zien, A. 2011. ℓ_p-norm multiple kernel learning. *Journal of Machine Learning Research* 12(Mar):953–997.

Kulis, B.; Saenko, K.; and Darrell, T. 2011. What you saw is not what you get: Domain adaptation using asymmetric kernel transforms. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, 1785–1792. IEEE.

Li, W.; Duan, L.; Xu, D.; and Tsang, I. W. 2014. Learning with augmented features for supervised and semi-supervised heterogeneous domain adaptation. *IEEE transactions on pattern analysis and machine intelligence* 36(6):1134–1148.

Li, B.; Yang, Q.; and 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*, 617–624. ACM.

Liu, H., and Yang, Y. 2015. Bipartite edge prediction via transductive learning over product graphs. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, 1880–1888.

Long, M.; Wang, J.; Ding, G.; Pan, S. J.; and Yu, P. S. 2014. Adaptation regularization: A general framework for transfer learning. *Knowledge and Data Engineering, IEEE Transactions on* 26(5):1076–1089.

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

Pan, S. J.; Tsang, I. W.; Kwok, J. T.; and Yang, Q. 2011. Domain adaptation via transfer component analysis. *Neural Networks, IEEE Transactions on* 22(2):199–210.

Prettenhofer, P., and Stein, B. 2010. Cross-language text classification using structural correspondence learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, 1118–1127. Association for Computational Linguistics.

Zhou, J. T.; Pan, S. J.; Tsang, I. W.; and Yan, Y. 2014. Hybrid heterogeneous transfer learning through deep learning. In *AAAI*, 2213–2220.

Zhu, Y.; Chen, Y.; Lu, Z.; Pan, S. J.; Xue, G.-R.; Yu, Y.; and Yang, Q. 2011. Heterogeneous transfer learning for image classification. In *AAAI*.

Zhu, X.; Lafferty, J.; and Rosenfeld, R. 2005. *Semi-supervised learning with graphs*. Carnegie Mellon University, language technologies institute, school of computer science.