Ji YS, Chen JJ, Niu G *et al.* Transfer learning via multi-view principal component analysis. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 26(1): 81–98 Jan. 2011. DOI 10.1007/s11390-011-1113-z

Transfer Learning via Multi-View Principal Component Analysis

Yang-Sheng Ji (吉阳生), Jia-Jun Chen (陈家骏), *Member*, *CCF*, Gang Niu (牛　罡)
Lin Shang (商　琳), *Member*, *CCF*, and Xin-Yu Dai (戴新宇), *Member*, *CCF*

Department of Computer Science and Technology, Nanjing University, Nanjing 210093, China
National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210093, China

E-mail: {jiys, chenjj}@nlp.nju.edu.cn; niugang@ai.nju.edu.cn; shanglin@nju.edu.cn; dxy@nlp.nju.edu.cn

Received December 31, 2009; revised November 12, 2010.

Abstract Transfer learning aims at leveraging the knowledge in labeled source domains to predict the unlabeled data in a target domain, where the distributions are different in domains. Among various methods for transfer learning, one kind of algorithms focus on the correspondence between bridge features and all the other specific features from different domains, and later conduct transfer learning via the *single-view* correspondence. However, the *single-view* correspondence may prevent these algorithms from further improvement due to the problem of incorrect correlation discovery. To tackle this problem, we propose a new method for transfer learning in a *multi-view* correspondence perspective, which is called *Multi-View Principal Component Analysis* (MVPCA) approach. MVPCA discovers the correspondence between bridge features representative across all domains and specific features from different domains respectively, and conducts the transfer learning by dimensionality reduction in a *multi-view* way, which can better depict the knowledge transfer. Experiments show that MVPCA can significantly reduce the cross domain prediction error of a baseline non-transfer method. With *multi-view* correspondence information incorporated to the *single-view* transfer learning method, MVPCA can further improve the performance of one state-of-the-art *single-view* method.

Keywords transfer learning, multi-view principal component analysis, text mining, sentiment classification

1 Introduction

In machine learning, traditional classification and regression algorithms strictly rely on the i.i.d. assumption, which requires that training and testing data are independent and identically distributed. However, real world applications usually show disobedience of this assumption. Let us take the sentiment polarity classification of product reviews on Amazon as an example. For one kind of product selling on Amazon, we can use all the labeled reviews to train a sentiment polarity classifier for the reviews of this product. However, new products are emerging on Amazon every day and lots of new product reviews will be posted afterwards. How could these newly posted product reviews be automatically classified as "positive" or "negative"? Of course, we can achieve this by spending lots of human efforts and time to annotate huge amount of new reviews to train a new classifier. But it is so expensive that we cannot do this all the time. Another easier way is to predict the sentiment polarity of new reviews by adapting an existing classifier trained on another domain. However, different types of product reviews tend to use different domain specific terms. So the term distributional difference will cause a lot of troubles for traditional classification methods. Table 1 shows some domain specific terms from four domains on Amazon: Book, DVD, Electronic, Kitchen. This gives us an intuitive impression on the distributional difference between domains. To solve this non-i.i.d. classification problem, transfer learning research has attracted a lot of interest.

Transfer learning tackles the problem of predicting

Table 1. Domain Specific Terms

Book	DVD	Electronic	Kitchen
two-stars	script	warranty	ease
must-read	dumb	it-worked	broken
pages	pathetic	unreliable	kitchen
repetitive	atrocious	works-great	convenient
great-book	great-dvd	stopped-working	cooking
poorly-written	predictable	memory	easy-to
excellent-book	intense	an-error	clean
required-reading	pointless	battery	sharp

Regular Paper

This work is supported by the National Natural Science Foundation of China under Grant No. 61003112, the National Basic Research 973 Program of China under Grant No. 2010CB327903, the National High-Tech Research and Development 863 Program of China under Grant No. 2006AA010109, and the Natural Science Foundation of Jiangsu Province under Grant No. 2009233.

testing instances drawn from a different but related distribution compared with training instances. Generally speaking, transfer learning aims at minimizing the differences between distributions of different domains to minimize the cross domain prediction error. In this paper, we assume that $P(\mathcal{X}_s) \neq P(\mathcal{X}_t)$ and $P(\mathcal{Y}_s) = P(\mathcal{Y}_t)$, which are in a common setting of transfer learning. $P(\mathcal{X}_s)$ and $P(\mathcal{Y}_s)$ denote the distributions of instances and labels in source domain respectively. $P(\mathcal{X}_t)$ and $P(\mathcal{Y}_t)$ are similarly defined in the target domain.

To achieve the goal of minimizing the prediction error across domains, transfer learning methods aim at making $P(\mathcal{X}_s)$ and $P(\mathcal{X}_t)$ less differ. Among proposed approaches, one kind of algorithms try to learn a low dimensional semantic space, which is able to depict the common knowledge across source and target domains. Represented by the common knowledge in the semantic space, source domain and target domain become more similar. In this way, Blitzer *et al.*[1-2] proposes a *single-view* transfer learning algorithm, Structural Correspondence Learning (SCL), in which a latent semantic space is generated via a *single-view* correspondence. SCL[1-2] seeks to learn the common knowledge across domains in the semantic space by applying the SVD procedure to a correspondence matrix. This correspondence matrix represents the statistical correlations between bridge features and all other features. Briefly, bridge features are those representative words occurring frequently across all domains, with detailed specification in Subsection 3.2. It is notable that this correspondence matrix mixes up all the other features across domains except bridge features, and measures the statistical correlations between them. In this paper, we regard this matrix as a *single-view* correspondence. *Single-view* correspondence can really help knowledge transfer. If the word "well-written" is a unique term in Book domain, and the word "use-easily" is another unique term in Kitchen domain, then they may never be correlated since they only occur separately. So if a classifier $f(\text{"well-written"})$ is trained on Book domain, it has no prediction power on the word "use-easily" in Kitchen domain. However, when you consider another bridge word "great", which may have strong correlations with both "well-written" and "use-easily", $f(\text{"well-written"})$ becomes predictive on the word "use-easily". Because when you see "well-written", it is assumed that "great" can be seen in some extend and then the probability of seeing the word "use-easily" can be estimated.

The above example shows how SCL[1-2] works. However, SCL has some limitations by utilizing the *single-view* correspondence. Let us zoom into the limitations by digesting a negative example. In DVD domain, if one product reviewer says "read-the-book", he means this DVD is "boring". So "read-the-book" has high correlation with "boring" in DVD domain. While taking a look at Book domain, if one product reviewer says "read-the-book", he means the book is "well-written" or "great". Surprisingly, besides the term "boring", "read-the-book" also has high correlations with "well-written" and "great". Obviously, it is a contradiction which is caused by the vague meanings of "read-the-book" in different domains. If we mix the instances from Book domain and DVD domain as SCL[1-2] does, and learn the *single-view* correspondence matrix on the mixed dataset, we can get a strong prediction between "boring" and "great". In this paper, we call this phenomenon incorrect correlation discovery. As these words with vague meanings do appear in sentiment texts on the Internet, we may often encounter incorrect correlations between words if we apply the *single-view* transfer learning methods. This problem would surely decrease the performances of these *single-view* algorithms.

In order to alleviate this problem, this paper proposes a new transfer learning method by utilizing *Multi-View Principal Component Analysis* (MVPCA). In MVPCA, firstly we choose m bridge features as SCL[1-2] does. Secondly, in every single domain, we learn a correspondence between bridge features and the other features. We call these correspondences in different domains *multi-view* correspondences. In this way, we separate the *single-view* correspondence of SCL algorithm into multiple parts. In every single view of correspondence of MVPCA, those words with vague meanings can only show its domain dependent meaning and they are not correlated with those contradictory words simultaneously in the single correspondence. Thus incorrect correlations between specific words from different domains are prevented. Finally, we apply MVPCA to learn a low dimensional semantic space. In this procedure, common knowledge of cross domain correlation is preserved in the eigenvectors as SCL does. Thus, knowledge transfer can be facilitated.

Compared with previous related work, the main contributions of MVPCA algorithm include the following.

1) This paper points out that words with vague meanings may do harm to *single-view* transfer learning methods. And we propose MVPCA to tackle this problem in a *multi-view* perspective. To the best of our knowledge, no previous work performs multi-view learning before prediction function $f(\cdot)$ is learned and no previous transfer learning method aims at this problem.

2) MVPCA opens a new scope for transfer learning. In order to learn a low dimensional semantic space,

multi-view correspondence learning offers a different way from *single-view* correspondence learning, which may achieve neat and clear correspondences between bridge features and different domain specific features in each domain. With the neat correspondence information incorporated into the *single-view* correspondence matrix, experimental results show that the performance of *single-view* transfer learning method can be further improved.

The rest of this paper is organized as follows. Section 2 discusses related work. Section 3 describes MVPCA algorithm in detail. Section 4 illustrates experimental results to show the effectiveness of MVPCA. Section 5 gives the conclusion and discussion on future work.

2 Related Work

Recently, many transfer learning algorithms have been proposed in the line of extracting common latent semantic space. Following this intuitive idea, SCL[1-2], Co-Clustering[3], and Transferred Component Analysis[4] are applied to learning the low dimensional semantic space. In these low dimensional semantic space learning methods, SCL[1-2] is one of the state-of-the-art transfer learning methods.

Daumé[5] suggests a new feature representation for both source domain and target domain. In this new feature space, both domain specific features and shared common features can be recognized to help improve knowledge transfer. Although [5] is easy to perform, it may suffer from "the curse of dimensionality". Daumé *et al.*[6] proposes a different way of transfer via adaptation on statistical maximal entropy classifier. Dai *et al.*[7] carries out the co-clustering[3] procedure on data combination of both source domain and target domain to find a shared latent space across domains. Similarly, Pan *et al.*[4] proposes a transferred component analysis to reduce the *Maximum Mean Discrepancy* (MMD) as a criterion to measure the difference between domains. All of the methods mentioned above achieve remarkable performances on knowledge transfer via newly generated latent space.

It is notable that Pan *et al.*[8] also propose a *multi-view* transfer learning algorithm, which predicts physical locations via an estimation function mapping from signal space to physical location space. Resulted from time variant signal space, *multi-view* refers to one view of original estimation function and the other view of current signal space data, which is utilized to adjust the previous estimation function to predict current location. MVPCA is different from [8]. On one hand, MVPCA performs *multi-view* learning before prediction function $f(\cdot)$ is estimated, while [8] performs *multi-view* adjustment after prediction function on source domain has been estimated. On the other hand, before the prediction function is estimated, MVPCA can employ as much information and as neat correspondence as it could to improve the transferability of the latent semantic space. The neat correspondence information may enable MVPCA to better depict intrinsic shared structures across domains than [8] does.

Accompanying with the viewpoint of extracting latent semantic space across domains, transfer learning algorithms aiming at instance weighting are proposed as well. Dai *et al.*[9] assign different weights to instances in different domains, and transfer the knowledge in a similar way of boosting. Huang *et al.*[10], Zadrozny[11] consider domain difference as a sample selection problem which differs the importance of instances in transfering. Jiang *et al.*[12] solve an application in natural language processing in the viewpoint of instance weighting. Besides above algorithms, Raina *et al.*[13] apply the sparse coding algorithm[14] to the problem of self-taught learning and achieves satisfactory performance. Tan *et al.*[15] and Dai *et al.*[16] propose modified versions of Naive Bayes to tackle the transfer learning in different viewpoints. Sandler *et al.*[17] incorporate manifold information to depict the relationship between features, and facilitate knowledge transfer via feature network constraint.

3 Transfer Learning via Multi-View Perspective

3.1 Problem Setting and Notations

Before introducing Multi-View Principal Component Analysis algorithm in detail, we should specify the problem setting and notations. In our transfer learning setting, *domain* and *task* are two basic concepts following the common transfer learning analysis.

A domain $\mathcal{D}$ consists of a feature space $\mathcal{X} \subset \mathbb{R}^d$ and label $\mathcal{Y} \subset \mathbb{R}$, distributions $P(\mathcal{X})$ and $P(\mathcal{Y})$, i.e., $\mathcal{D} = \{\mathcal{X}, \mathcal{Y}, P(\mathcal{X}), P(\mathcal{Y})\}$. Also $\mathcal{D}$ can be rewritten as $\mathcal{D} = \{(\boldsymbol{x}_i, y_i) | \boldsymbol{x}_i \in \mathcal{X}, y_i \in \mathcal{Y}\}$, which implies $P(\mathcal{X})$ and $P(\mathcal{Y})$ implicitly; $\boldsymbol{x}_i$ denotes an instance in domain $\mathcal{D}$ which is a term vector of a document with Bag of Words or TFIDF representation; y_i denotes the label of instance $\boldsymbol{x}_i$; $[\boldsymbol{x}_i]_j$ denotes the j-th coordinate value of the instance and $[\boldsymbol{x}_i]_j \in \mathcal{X}_j$. Source domain is denoted by $\mathcal{D}_{\mathrm{s}} = \mathcal{D}_{\mathrm{sl}} \bigcup \mathcal{D}_{\mathrm{su}} \subseteq \mathcal{X}_{\mathrm{s}} \times \mathcal{Y}_{\mathrm{s}}$, $\mathcal{Y}_{\mathrm{s}} = \{+1, -1, 0\}$. $\mathcal{D}_{\mathrm{sl}} = \{(\boldsymbol{x}_i^{\mathrm{sl}}, y_i^{\mathrm{sl}}) | i = 1, \ldots, n_{\mathrm{sl}}\}$ denotes labeled data; $\mathcal{D}_{\mathrm{su}} = \{(\boldsymbol{x}_i^{\mathrm{su}}, y_i^{\mathrm{su}} = 0) | i = 1, \ldots, n_{\mathrm{su}}\}$ denotes unlabeled data; $\boldsymbol{X}_{\mathrm{s}} = \{\boldsymbol{x}_i^{\mathrm{s}} | i = 1, \ldots, n_{\mathrm{sl}} + n_{\mathrm{su}}\}$; $\boldsymbol{Y}_{\mathrm{s}} = \{y_i^{\mathrm{s}} | i = 1, \ldots, n_{\mathrm{sl}} + n_{\mathrm{su}}\}$; $\boldsymbol{x}_i^{\mathrm{s}}$ is a positive\negative instance if $y_i^{\mathrm{s}} = +1 \setminus -1$, $\boldsymbol{x}_i^{\mathrm{s}}$ is an unlabeled instance if $y_i^{\mathrm{s}} = 0$. $\mathcal{D}_{\mathrm{t}}$, $\mathcal{D}_{\mathrm{tl}}$, $\mathcal{D}_{\mathrm{tu}}$, $\boldsymbol{X}_{\mathrm{t}}$ and $\boldsymbol{Y}_{\mathrm{t}}$ are similarly defined in target domain.

A task $\mathcal{T}$ consists of a specified domain $\mathcal{D}$ and a prediction function $f(\cdot)$, i.e., $\mathcal{T} = \{\mathcal{D}, f(\cdot)\}$. $f(\cdot)$ is a prediction function learned on domain $\mathcal{D}$, which maximizes likelihood or minimizes a loss function, i.e., $f(\cdot) = \arg\min_f \sum_{\boldsymbol{x}_i \in \boldsymbol{X}} \|f(\boldsymbol{x}_i) - h(\boldsymbol{x}_i)\|_2^2$, where $h(\cdot)$ is a ground truth prediction function on domain $\mathcal{D}$.

Now we can give a formal definition of transfer learning in our setting.

Definition 1 (Transfer Learning). *Given source domain $\mathcal{D}_s = \mathcal{D}_{sl} \bigcup \mathcal{D}_{su}$ and target domain $\mathcal{D}_t = \mathcal{D}_{tu}$, transfer learning aims at learning a task $\mathcal{T} = \{\mathcal{D}_s \bigcup \mathcal{D}_t, f(\cdot)\}$ in order that $err_{\mathcal{D}_t}(f) = \sum_{\boldsymbol{x}_i \in \boldsymbol{X}_t} \|f(\boldsymbol{x}_i) - h_t(\boldsymbol{x}_i)\|_2^2$ can be minimized. Note that, $h_t(\cdot)$ is a ground truth prediction function on domain $\mathcal{D}_t$, and we often approximate $h_t(\boldsymbol{x}_i^t)$ by y_i^t. This definition holds under following assumptions*:

1) $\mathcal{X}_s \bigcap \mathcal{X}_t \neq \varnothing$;

2) $P(\mathcal{X}_s) \neq P(\mathcal{X}_t)$;

3) $P(\mathcal{Y}_s) = P(\mathcal{Y}_t)$;

4) *this paper only focuses on one source domain and one target domain.*

3.2 Bridge Features and Multi-View Correspondence Learning

Let us come to the definition of bridge feature, which is one of the key parts of MVPCA algorithm. We denote one single bridge feature as $\mathcal{X}_{\Lambda_i}$, and denote the bridge feature set as $\mathcal{X}_\Lambda$. One of the most important reasons that MVPCA shows good performance on domain adaptation is that our chosen bridge features contribute great discriminative ability across domains when performing multi-view transfer learning. The discriminative power of bridge features is definitely derived from high correlations with class labels. So bridge features can be regarded as pseudo labels, and these discriminative pseudo labels can bring a lot of helpful discriminative power across domains to benefit the knowledge transfer.

For short, to achieve the discriminative power on both domains, bridge features must satisfy the following assumptions:

• $\mathcal{X}_{\Lambda_i}$ is one of the representative features in both domains; all bridge features constitute bridge feature set $\mathcal{X}_\Lambda = \{\mathcal{X}_{\Lambda_1}, \ldots, \mathcal{X}_{\Lambda_m}\} \subseteq \mathcal{X}_s \bigcap \mathcal{X}_t$;

• each bridge feature $\mathcal{X}_{\Lambda_i}$ is statistical correlated with class labels. In this paper, mutual information is utilized as the measurement.

To construct the bridge feature set $\mathcal{X}_\Lambda$ in MVPCA, we may simply pick up m features $\mathcal{X}_i \subseteq \mathcal{X}_{\mathrm{s}} \bigcap \mathcal{X}_{\mathrm{t}}$ with the highest mutual information values and group them into $\mathcal{X}_\Lambda$. Mutual information is computed between feature $\mathcal{X}_i$ and $\mathcal{Y}_s$ on $\mathcal{D}_{\mathrm{sl}}$ according to (1):

$$I(\mathcal{X}_i; \mathcal{Y}_s) = \sum_{y \in \mathcal{Y}_s} \sum_{x \in \mathcal{X}_i} p(x, y) \log \frac{p(x, y)}{p_1(x)\, p_2(y)}. \quad (1)$$

After $\mathcal{X}_\Lambda$ is ready, *multi-view* correspondence can be extracted thereafter. In this paper, *multi-view* correspondence consists of two views on source and target domains respectively. One view is the correspondence between bridge features $\mathcal{X}_\Lambda$ and other features $\mathcal{X}_{\mathrm{s}} - \mathcal{X}_\Lambda$ in $\mathcal{D}_{\mathrm{s}}$; the other view is the correspondence between bridge features and other features $\mathcal{X}_{\mathrm{t}} - \mathcal{X}_\Lambda$ in $\mathcal{D}_{\mathrm{t}}$. Anyway, *two-view* MVPCA algorithm is a straightforward and representative version of *multi-view* MVPCA, and the *multi-view* version can be easily extended, which is specified in Theorem 2.

Then let us take source domain $\mathcal{D}_{\mathrm{s}}$ as an example to illustrate how to measure the correspondence between bridge feature $\mathcal{X}_{\Lambda_i}$ and other text features in $\mathcal{X}_{\mathrm{s}}$ in detail. For each instance $(\boldsymbol{x}_i^{\mathrm{su}}, y_i^{\mathrm{su}}) \in \mathcal{D}_{\mathrm{su}}$, check the value of bridge feature $\mathcal{X}_{\Lambda_i}$ in instance $(\boldsymbol{x}_i^{\mathrm{su}}, y_i^{\mathrm{su}})$. If $[\boldsymbol{x}_i^{\mathrm{su}}]_{\Lambda_i} = 0$, set $y_i^{\mathrm{su}} = -1$; otherwise, set $[\boldsymbol{x}_i^{\mathrm{su}}]_{\Lambda_i} = 0$, $y_i^{\mathrm{su}} = +1$. In this way, we can construct a pseudo labeled dataset $\mathcal{D}_{\mathrm{s}}^{\Lambda_i}$. On this pseudo labeled domain, a linear classifier $f_{\mathcal{D}_{\mathrm{s}}^{\Lambda_i}}(\cdot)$ can be learned and $f_{\mathcal{D}_{\mathrm{s}}^{\Lambda_i}}(\cdot)$ provides a classification hyper plane $\boldsymbol{\theta}_{\mathcal{D}_{\mathrm{s}}^{\Lambda_i}} \in \mathbb{R}^{d_{\mathrm{s}}}$. This hyper plane $\boldsymbol{\theta}_{\mathcal{D}_{\mathrm{s}}^{\Lambda_i}}$ can be regarded as the correspondence between bridge feature $\mathcal{X}_{\Lambda_i}$ and other features on source domain. So we may arrange all $\boldsymbol{\theta}_{\mathcal{D}_{\mathrm{s}}^{\Lambda_i}}$ together as $\boldsymbol{\Theta}_{\mathcal{D}_{\mathrm{s}}^{\Lambda}} = [\boldsymbol{\theta}_{\mathcal{D}_{\mathrm{s}}^{\Lambda_1}}, \ldots, \boldsymbol{\theta}_{\mathcal{D}_{\mathrm{s}}^{\Lambda_m}}] \in \mathbb{R}^{d_{\mathrm{s}} \times m}$. Similarly, $\boldsymbol{\Theta}_{\mathcal{D}_{\mathrm{t}}^{\Lambda}} = [\boldsymbol{\theta}_{\mathcal{D}_{\mathrm{t}}^{\Lambda_1}}, \ldots, \boldsymbol{\theta}_{\mathcal{D}_{\mathrm{t}}^{\Lambda_m}}] \in \mathbb{R}^{d_{\mathrm{t}} \times m}$ can be constructed in the same way. Up to now, every bridge feature $\mathcal{X}_{\Lambda_i}$ has set up *multi-view* correspondences with features in source domain and target domain respectively: $\boldsymbol{\theta}_{\mathcal{D}_{\mathrm{s}}^{\Lambda_i}}$ and $\boldsymbol{\theta}_{\mathcal{D}_{\mathrm{t}}^{\Lambda_i}}$. By putting them together, $(\boldsymbol{\theta}_{\mathcal{D}_{\mathrm{s}}^{\Lambda_i}}^{\mathrm{T}}, \boldsymbol{\theta}_{\mathcal{D}_{\mathrm{t}}^{\Lambda_i}}^{\mathrm{T}})$ represents a *multi-view* perspective on the correspondences of bridge feature $\mathcal{X}_{\Lambda_i}$. So we rewrite $\boldsymbol{\Theta}_{\mathcal{D}_{\mathrm{s}}^{\Lambda}}$ and $\boldsymbol{\Theta}_{\mathcal{D}_{\mathrm{t}}^{\Lambda}}$ as $\boldsymbol{\Theta}_{\Lambda}^{(\mathrm{s})}$ and $\boldsymbol{\Theta}_{\Lambda}^{(\mathrm{t})}$ respectively to show that they are two views of feature correspondences.

By setting up *multi-view* correspondence on both domains, strong correlations between bridge features and specific features can be detected within every domain. In this way, those words with vague meanings can only give one sentimental polarity in one view of the correspondence. And those contradictory correlations are prevented in correspondence. As a result, *multi-view* correspondences can give a cleaner and neater within domain knowledge, and preserve cross domain knowledge via bridge features as SCL does. To conclude this subsection, in *multi-view* perspective, we separate the correspondence between features from *single-view* to *multi-view* in order to better depict within domain

knowledge while preserving the same cross domain knowledge as SCL does.

3.3 Multi-View Principal Component Analysis

After setting up the *multi-view* correspondence, we may apply Multi-View Principal Component Analysis to these correspondence matrices to get a latent semantic space, which consists of a group of eigenvectors. These eigenvectors of MVPCA not only prevent incorrect correlation generation, but also preserve the cross domain knowledge. Thus MVPCA can further improve the performance of domain adaptation of SCL.

3.3.1 Objective Function Specification and Notations

Before introducing the details of MVPCA algorithm, we specify the intuitive idea and notations first. Suppose we have *two views* of instances $\boldsymbol{X}_{\mathrm{s}} = \{\boldsymbol{x}_i^{\mathrm{s}} \in \mathbb{R}^{d_{\mathrm{s}}} | i = 1, \ldots, m\}$ and $\boldsymbol{X}_{\mathrm{t}} = \{\boldsymbol{x}_i^{\mathrm{t}} \in \mathbb{R}^{d_{\mathrm{t}}} | i = 1, \ldots, m\}$. We call $\boldsymbol{X}_{\mathrm{s}}$ and $\boldsymbol{X}_{\mathrm{t}}$ source view and target view respectively. And the instances with index i in source view and target view, e.g., $\boldsymbol{x}_i^{\mathrm{s}}$ and $\boldsymbol{x}_i^{\mathrm{t}}$, are describing the same instance. So all the m instances in both views correspond one to one.

In order to generate a low dimensional semantic space, one common way is to compress instances from *two views* into *single view* $\boldsymbol{X} = \{\boldsymbol{x}_i \in \mathbb{R}^{d_{\mathrm{s}}+d_{\mathrm{t}}} | \boldsymbol{x}_i = \langle \boldsymbol{x}_i^{\mathrm{s}}, \boldsymbol{x}_i^{\mathrm{t}} \rangle;\ i = 1, \ldots, m\}$. $\boldsymbol{x}_i = \langle \boldsymbol{x}_i^{\mathrm{s}}, \boldsymbol{x}_i^{\mathrm{t}} \rangle$ denotes the spliced vector of $\boldsymbol{x}_i^{\mathrm{s}}$ and $\boldsymbol{x}_i^{\mathrm{t}}$. Then by applying SVD or PCA on $\boldsymbol{X}$, we can get a low dimensional semantic space. However, there are some drawbacks of this method. Firstly, when applied to transfer learning, we can still see the incorrect correlation between some contradictory domain specific features. So in the low dimensional semantic space, helpful common knowledge and incorrect correlations are fused together, which can do harm to the transfer learning. As a result, we cannot expect it to show better performance than the *single-view* transfer learning method. Secondly, Vinokourov *et al.*[18] and Li *et al.*[19] point out that when performing cross-language classification, this naive method does not have promising results as Hardoon's algorithm[20] does.

In another way, if we can learn a set of uniform orthogonal eigenvectors $\boldsymbol{U} = \{\boldsymbol{u}_j | j = 1, \ldots, p\}$ across different feature spaces of $\mathcal{X}_{\mathrm{s}}$ and $\mathcal{X}_{\mathrm{t}}$, the set of eigenvectors $\boldsymbol{U}$ may not only preserve helpful common knowledge across source and target domains, but also prevent incorrect correlation generation. While learning a set of unified eigenvectors across different feature spaces seems infeasible, we can achieve this by merging $\mathcal{X}_{\mathrm{s}}$ and $\mathcal{X}_{\mathrm{t}}$ into a unified feature space $\hat{\mathcal{X}} = \hat{\mathcal{X}}_{\mathrm{s}} = \hat{\mathcal{X}}_{\mathrm{t}} \subset \mathbb{R}^{\mathrm{d}}$, which is just the same as a merged dictionary from another two. In the unified feature space, $\boldsymbol{X}_{\mathrm{s}} = \{\boldsymbol{x}_i^{\mathrm{s}} \in \mathbb{R}^{d_{\mathrm{s}}} | i = 1, \ldots, m\}$ and $\boldsymbol{X}_{\mathrm{t}} = \{\boldsymbol{x}_i^{\mathrm{t}} \in \mathbb{R}^{d_{\mathrm{t}}} | i = 1, \ldots, m\}$ are rewritten as $\hat{\boldsymbol{X}}_{\mathrm{s}} = \{\hat{\boldsymbol{x}}_i^{\mathrm{s}} \in \mathbb{R}^{d} | i = 1, \ldots, m\}$ and $\hat{\boldsymbol{X}}_{\mathrm{t}} = \{\hat{\boldsymbol{x}}_i^{\mathrm{t}} \in \mathbb{R}^{d} | i = 1, \ldots, m\}$. We may also denote $\hat{\boldsymbol{X}}_{\mathrm{s}}$ by $[\hat{\boldsymbol{x}}_1^{\mathrm{s}}, \ldots, \hat{\boldsymbol{x}}_m^{\mathrm{s}}]$ and denote $\hat{\boldsymbol{X}}_{\mathrm{t}}$ by $[\hat{\boldsymbol{x}}_1^{\mathrm{t}}, \ldots, \hat{\boldsymbol{x}}_m^{\mathrm{t}}]$ for convenience. Under these settings, we give the details of Multi-View Principal Component Analysis:

Definition 2 (Objective Function of Multi-View Principal Component Analysis). *Suppose we have two views of instances $\boldsymbol{X}_s = \{\boldsymbol{x}_i^s \in \mathbb{R}^{d_s} | i = 1, \ldots, m\}$ and $\boldsymbol{X}_t = \{\boldsymbol{x}_i^t \in \mathbb{R}^{d_t} | i = 1, \ldots, m\}$. Given a parameter p, Multi-View Principal Component Analysis tries to find a set of p uniform orthogonal vectors $\boldsymbol{U} = \{\boldsymbol{u}_1, \ldots, \boldsymbol{u}_p\}$ which minimizes:*

$$\min_{\boldsymbol{U}} \mathcal{E}(\hat{\boldsymbol{X}}_{\mathrm{s}}, \hat{\boldsymbol{X}}_{\mathrm{t}}, \boldsymbol{U}, p) = \sum_{i=1}^{m} (\mathcal{E}_{\mathrm{s}}^2(\hat{\boldsymbol{x}}_i^{\mathrm{s}}, \boldsymbol{U}, p) + \mathcal{E}_{\mathrm{t}}^2(\hat{\boldsymbol{x}}_i^{\mathrm{t}}, \boldsymbol{U}, p)) \tag{2}$$

$$\begin{aligned} s.t. \quad \mathcal{E}_{\mathrm{s}}(\hat{\boldsymbol{x}}_i^{\mathrm{s}}, \boldsymbol{U}, p) &= \hat{\boldsymbol{x}}_i^{\mathrm{s}} - \sum_{j=1}^{p} ((\hat{\boldsymbol{x}}_i^{\mathrm{s}})^{\mathrm{T}} \cdot \boldsymbol{u}_j) \boldsymbol{u}_j \\ &= \hat{\boldsymbol{x}}_i^{\mathrm{s}} - \sum_{j=1}^{p} \boldsymbol{u}_j (\hat{\boldsymbol{x}}_i^{\mathrm{s}})^{\mathrm{T}} \boldsymbol{u}_j \end{aligned} \tag{3}$$

$$\begin{aligned} \mathcal{E}_t(\hat{\boldsymbol{x}}_i^{t}, \boldsymbol{U}, p) &= \hat{\boldsymbol{x}}_i^{t} - \sum_{j=1}^{p} ((\hat{\boldsymbol{x}}_i^{t})^{\mathrm{T}} \cdot \boldsymbol{u}_j) \boldsymbol{u}_j \\ &= \hat{\boldsymbol{x}}_i^{t} - \sum_{j=1}^{p} \boldsymbol{u}_j (\hat{\boldsymbol{x}}_i^{t})^{\mathrm{T}} \boldsymbol{u}_j \end{aligned} \tag{4}$$

$$\boldsymbol{u}_i^{\mathrm{T}} \boldsymbol{u}_j = \begin{cases} 1, & \text{if } i = j, \\ 0, & \text{if } i \neq j. \end{cases} \tag{5}$$

The set of p uniform orthogonal vectors $\boldsymbol{U} = \{\boldsymbol{u}_1, \ldots, \boldsymbol{u}_p\}$ build up a low dimensional semantic space. And in this semantic space, every instance can be reconstructed by the new coordinate system. The difference between original instance and reconstructed one is called residual error, denoted by $\mathcal{E}_{\mathrm{s}}^2(\hat{\boldsymbol{x}}_i^{\mathrm{s}}, \boldsymbol{U}, p)$ and $\mathcal{E}_{\mathrm{t}}^2(\hat{\boldsymbol{x}}_i^{\mathrm{t}}, \boldsymbol{U}, p)$. So MVPCA aims at finding the semantic space which minimizes the total residual error of each instance under all views.

Theorem 1. *The first p eigenvectors with the largest eigenvalues of $[\hat{\boldsymbol{X}}_s, \hat{\boldsymbol{X}}_t][\hat{\boldsymbol{X}}_s, \hat{\boldsymbol{X}}_t]^{\mathrm{T}}$ give the solution of the uniform orthogonal vector set to Definition 2.*

Proof. Firstly, let

$$\mathcal{E}(\hat{\boldsymbol{X}}_{\mathrm{s}}, \boldsymbol{U}, p) = \sum_{i=1}^{m} \mathcal{E}_{\mathrm{s}}^2(\hat{\boldsymbol{x}}_i^{\mathrm{s}}, \boldsymbol{U}, p) \tag{6}$$

$$= \sum_{i=1}^{m} (\hat{\boldsymbol{x}}_i^{\mathrm{s}} - \sum_{j=1}^{p} \boldsymbol{u}_j (\hat{\boldsymbol{x}}_i^{\mathrm{s}})^{\mathrm{T}} \boldsymbol{u}_j)^2 \tag{7}$$

$$= \sum_{i=1}^{m} ((\hat{\boldsymbol{x}}_i^{\mathrm{s}})^{\mathrm{T}} (\hat{\boldsymbol{x}}_i^{\mathrm{s}}) - \sum_{j=1}^{p} \boldsymbol{u}_j^{\mathrm{T}} (\hat{\boldsymbol{x}}_i^{\mathrm{s}}) (\hat{\boldsymbol{x}}_i^{\mathrm{s}})^{\mathrm{T}} \boldsymbol{u}_j) \tag{8}$$

$$= trace(\hat{\boldsymbol{X}}_{\mathrm{s}}^{\mathrm{T}}\hat{\boldsymbol{X}}_{\mathrm{s}}) - \sum_{j=1}^{p}\boldsymbol{u}_j^{\mathrm{T}}\hat{\boldsymbol{X}}_{\mathrm{s}}\hat{\boldsymbol{X}}_{\mathrm{s}}^{\mathrm{T}}\boldsymbol{u}_j. \quad (9)$$

Similarly,

$$\mathcal{E}(\hat{\boldsymbol{X}}_{\mathrm{t}}, \boldsymbol{U}, p) = trace(\hat{\boldsymbol{X}}_{\mathrm{t}}^{\mathrm{T}}\hat{\boldsymbol{X}}_{\mathrm{t}}) - \sum_{j=1}^{p}\boldsymbol{u}_j^{\mathrm{T}}\hat{\boldsymbol{X}}_{\mathrm{t}}\hat{\boldsymbol{X}}_{\mathrm{t}}^{\mathrm{T}}\boldsymbol{u}_j. \quad (10)$$

Secondly,

$$\begin{aligned}
&\min_{\boldsymbol{U}} \mathcal{E}(\hat{\boldsymbol{X}}_{\mathrm{s}}, \hat{\boldsymbol{X}}_{\mathrm{t}}, \boldsymbol{U}, p) \\
&= \min_{\boldsymbol{U}}(\mathcal{E}(\hat{\boldsymbol{X}}_{\mathrm{s}}, \boldsymbol{U}, p) + \mathcal{E}(\hat{\boldsymbol{X}}_{\mathrm{t}}, \boldsymbol{U}, p)) \quad (11)\\
&= \min_{\boldsymbol{U}}\{trace(\hat{\boldsymbol{X}}_{\mathrm{s}}^{\mathrm{T}}\hat{\boldsymbol{X}}_{\mathrm{s}} + \hat{\boldsymbol{X}}_{\mathrm{t}}^{\mathrm{T}}\hat{\boldsymbol{X}}_{\mathrm{t}}) - \\
&\quad \sum_{j=1}^{p}\boldsymbol{u}_j^{\mathrm{T}}(\hat{\boldsymbol{X}}_{\mathrm{s}}\hat{\boldsymbol{X}}_{\mathrm{s}}^{\mathrm{T}} + \hat{\boldsymbol{X}}_{\mathrm{t}}\hat{\boldsymbol{X}}_{\mathrm{t}}^{\mathrm{T}})\boldsymbol{u}_j\} \quad (12)\\
&= \max_{\boldsymbol{U}}\sum_{j=1}^{p}\boldsymbol{u}_j^{\mathrm{T}}(\hat{\boldsymbol{X}}_{\mathrm{s}}\hat{\boldsymbol{X}}_{\mathrm{s}}^{\mathrm{T}} + \hat{\boldsymbol{X}}_{\mathrm{t}}\hat{\boldsymbol{X}}_{\mathrm{t}}^{\mathrm{T}})\boldsymbol{u}_j \quad (13)\\
&= \max_{\boldsymbol{U}}\sum_{j=1}^{p}\boldsymbol{u}_j^{\mathrm{T}}[\hat{\boldsymbol{X}}_{\mathrm{s}}, \hat{\boldsymbol{X}}_{\mathrm{t}}][\hat{\boldsymbol{X}}_{\mathrm{s}}, \hat{\boldsymbol{X}}_{\mathrm{t}}]^{\mathrm{T}}\boldsymbol{u}_j. \quad (14)
\end{aligned}$$

From the last step, we can see that $\max_{\boldsymbol{U}}\sum_{j=1}^{p}\boldsymbol{u}_j^{\mathrm{T}}[\hat{\boldsymbol{X}}_{\mathrm{s}}, \hat{\boldsymbol{X}}_{\mathrm{t}}][\hat{\boldsymbol{X}}_{\mathrm{s}}, \hat{\boldsymbol{X}}_{\mathrm{t}}]^{\mathrm{T}}\boldsymbol{u}_j$ is a standard form of Principal Component Analysis, and the solution to $\boldsymbol{U}$ consists of the first p eigenvectors with largest eigenvalues of matrix $[\hat{\boldsymbol{X}}_{\mathrm{s}}, \hat{\boldsymbol{X}}_{\mathrm{t}}][\hat{\boldsymbol{X}}_{\mathrm{s}}, \hat{\boldsymbol{X}}_{\mathrm{t}}]^{\mathrm{T}}$. □

If we have multiple views $\boldsymbol{X}_k = \{\boldsymbol{x}_i^k \in \mathbb{R}^{d_k} | i = 1, \ldots, m\}$, $k = 1, \ldots, K$, Theorem 1 can be further extended to a *multi-view* setting. After merging their feature spaces to a unified space, these multiple views are rewritten as $\hat{\boldsymbol{X}}_k = \{\hat{\boldsymbol{x}}_i^k \in \mathbb{R}^d | i = 1, \ldots, m\}$, with $k = 1, \ldots, K$.

Theorem 2. *If we have K views of instances, $\hat{\boldsymbol{X}}_1 = \{\hat{\boldsymbol{x}}_i^1 \in \mathbb{R}^d | i = 1, \ldots, m\}$ is the first view, with $\hat{\boldsymbol{X}}_k$ $(k = 2, \ldots, K)$ in the same notations. Given a parameter p, Multi-View Principal Component Analysis tries to find a set of p uniform orthogonal vectors $\boldsymbol{U} = \{\boldsymbol{u}_1, \ldots, \boldsymbol{u}_p\}$, which minimizes the residual error across all the views of instances*:

$$\min_{\boldsymbol{U}} \mathcal{E}(\hat{\boldsymbol{X}}_1, \ldots, \hat{\boldsymbol{X}}_K, \boldsymbol{U}, p) = \sum_{i=1}^{m}\sum_{k=1}^{K}(\mathcal{E}_k^2(\hat{\boldsymbol{x}}_i^k, \boldsymbol{U}, p)) \quad (15)$$

$$\text{s.t.} \quad \mathcal{E}_k(\hat{\boldsymbol{x}}_i^k, \boldsymbol{U}, p) = \hat{\boldsymbol{x}}_i^k - \sum_{j=1}^{p}\boldsymbol{u}_j(\hat{\boldsymbol{x}}_i^k)^{\mathrm{T}}\boldsymbol{u}_j, \ k = 1, \ldots, K \quad (16)$$

$$\boldsymbol{u}_i^{\mathrm{T}}\boldsymbol{u}_j = \begin{cases} 1, & \text{if } i = j, \\ 0, & \text{if } i \neq j. \end{cases} \quad (17)$$

The first p eigenvectors with the largest eigenvalues of $[\hat{\boldsymbol{X}}_1, \ldots, \hat{\boldsymbol{X}}_K][\hat{\boldsymbol{X}}_1, \ldots, \hat{\boldsymbol{X}}_K]^{\mathrm{T}}$ give the solution.

Proof. This proof is rather similar to the proof of Theorem 1. We simply omit it here. □

3.3.2 Multi-View Principal Component Analysis Algorithm

When we apply *Multi-View Principal Component Analysis* to transfer learning, we need to replace $\boldsymbol{X}_s$ and $\boldsymbol{X}_t$ in Theorem 1 by $\boldsymbol{\Theta}_{\Lambda}^{(\mathrm{s})}$ and $\boldsymbol{\Theta}_{\Lambda}^{(\mathrm{t})}$ respectively and then we get the algorithm. Now let us move to the details of *Multi-View Principal Component Analysis* algorithm, which are shown in Table 2.

Table 2. Multi-View Principal Component Analysis Algorithm

Input: $\mathcal{D}_{\mathrm{s}} = \mathcal{D}_{\mathrm{sl}} \bigcup \mathcal{D}_{\mathrm{su}}$, $\mathcal{D}_{\mathrm{t}} = \mathcal{D}_{\mathrm{tu}}$.
Output: $\widehat{f}_{\mathcal{D}_{\mathrm{sl}}}$.
Begin
(1) Choose m features from $\mathcal{X}_{\mathrm{s}} \bigcap \mathcal{X}_{\mathrm{t}}$ to form bridge feature set $\mathcal{X}_{\Lambda}$.
(2) **For** $\mathcal{X}_{\Lambda_i}$ in $\mathcal{X}_{\Lambda}$
• construct pseudo labeled datasets: $\mathcal{D}_{\mathrm{s}}^{\Lambda_i}$ and $\mathcal{D}_{\mathrm{t}}^{\Lambda_i}$
• train linear classifiers on them: $f_{\mathcal{D}_{\mathrm{s}}^{\Lambda_i}}$ and $f_{\mathcal{D}_{\mathrm{t}}^{\Lambda_i}}$
• add $\boldsymbol{\theta}_{\Lambda_i}^{(\mathrm{s})}$ into $\boldsymbol{\Theta}_{\Lambda}^{(\mathrm{s})}$, $\boldsymbol{\theta}_{\Lambda_i}^{(\mathrm{t})}$ into $\boldsymbol{\Theta}_{\Lambda}^{(\mathrm{t})}$
End
(3) Run MVPCA:
$\boldsymbol{\mathcal{P}}_{\Lambda} = \mathrm{MVPCA}([\boldsymbol{\Theta}_{\Lambda}^{(\mathrm{s})}, \boldsymbol{\Theta}_{\Lambda}^{(\mathrm{t})}])$.
(4) $\widehat{\mathcal{D}}_{\mathrm{sl}} = \mathit{FeatureExpansion}(\mathcal{D}_{\mathrm{sl}}, \boldsymbol{P}_{\Lambda})$
$\widehat{\mathcal{D}}_{\mathrm{t}} = \mathit{FeatureExpansion}(\mathcal{D}_{\mathrm{t}}, \boldsymbol{\mathcal{P}}_{\Lambda})$
(5) Train a linear classifier $\widehat{f}_{\mathcal{D}_{\mathrm{sl}}}$ on $\widehat{\mathcal{D}}_{\mathrm{sl}}$.
$\widehat{f}_{\mathcal{D}_{\mathrm{sl}}}$ is the output.

Bridge features are those representative words occurring frequently across domains, which can be easily detected according to the specification in Subsection 3.2.

Then let us take source domain $\mathcal{D}_{\mathrm{s}}$ as an example to illustrate how to measure the correspondence matrices $\boldsymbol{\Theta}_{\Lambda}^{(\mathrm{s})}$ and $\boldsymbol{\Theta}_{\Lambda}^{(\mathrm{t})}$. For every bridge feature $\mathcal{X}_{\Lambda_i}$, we can construct a pseudo labeled dataset $\mathcal{D}_{\mathrm{s}}^{\Lambda_i}$. Next step is to train a linear classifier $f_{\mathcal{D}_{\mathrm{s}}^{\Lambda_i}}$ on this pseudo labeled domain. In this paper, logistic regression is utilized as our linear classier:

$$L(X, \lambda) = \max_{\boldsymbol{\theta}} \sum_{x \in X} \frac{1}{1 + e^{-y\boldsymbol{\theta}^{\mathrm{T}}x}} - \lambda||\boldsymbol{\theta}||_2^2. \quad (18)$$

We denote the hyper plane $\boldsymbol{\theta}$ of logistic regression as $\boldsymbol{\theta}_{\Lambda_i}^{(\mathrm{s})}$ on the pseudo labeled dataset $\mathcal{D}_{\mathrm{s}}^{\Lambda_i}$. Then we may arrange all $\boldsymbol{\theta}_{\Lambda_i}^{(\mathrm{s})}$ together as $\boldsymbol{\Theta}_{\Lambda}^{(\mathrm{s})}$, which is one view of feature correspondence. Similarly, $\boldsymbol{\Theta}_{\Lambda}^{(\mathrm{t})}$ is another view of feature correspondence. The following step is to run *Multi-View Principal Component Analysis* to get a projection matrix.

In feature generation phase, for instance $(\boldsymbol{x}_i^{\mathrm{s}}, y_i^{\mathrm{s}}) \in \mathcal{D}_{\mathrm{s}}$, $\boldsymbol{x}_i^{\mathrm{s}}$ is expanded to $\widehat{\boldsymbol{x}}_i^{s} = \langle \boldsymbol{x}_i^{\mathrm{s}}, \boldsymbol{\mathcal{P}}_{\Lambda}^{\mathrm{T}} \boldsymbol{x}_i^{\mathrm{s}} \rangle$. So in this feature expansion way, we get a feature expanded labeled data $\widehat{\mathcal{D}}_{\mathrm{sl}}$ for source domain. Similarly, we can get $\widehat{\mathcal{D}}_{\mathrm{t}}$. Finally, we train a classifier $\widehat{f}_{\mathcal{D}_{\mathrm{sl}}}$ on source domain $\widehat{\mathcal{D}}_{\mathrm{sl}}$, then we apply classifier $\widehat{f}_{\mathcal{D}_{\mathrm{sl}}}$ to predict new instances from target domain $\widehat{\mathcal{D}}_{\mathrm{t}}$.

What is more interesting, by incorporating *single-view* correspondences of SCL algorithm into MVPCA, MVPCA's performance can be further improved via learning a semantic space from both *multi-view* and *single-view* perspectives. This variant version of MVPCA is presented in Table 3, and we denote it by Variant Multi-View Principal Component Analysis (VMVPCA), which can also be regarded as a combined algorithm of MVPCA and SCL.

On conclusion of this section, we briefly review Multi-View Principal Component Analysis method. MVPCA separates *single-view* correspondence between features into *multiple views* of feature correspondence. The separate *multiple views* prevent the generation of incorrect and contradictory correlation between domain specific features. After that, we apply Multi-View Principal Component Analysis to generate a semantic space which preserves common knowledge across domains. In this way, MVPCA algorithm overcomes the drawback of *single-view* transfer learning methods, and preserves the benefits for knowledge transfer.

Table 3. Variant Multi-View Principal Component Analysis Algorithm

Input: $\mathcal{D}_{\mathrm{s}} = \mathcal{D}_{\mathrm{sl}} \bigcup \mathcal{D}_{\mathrm{su}}$, $\mathcal{D}_{\mathrm{t}} = \mathcal{D}_{\mathrm{tu}}$.

Output: $\widehat{f}_{\mathcal{D}_{\mathrm{sl}}}$.

Begin

(1) Choose m features from $\mathcal{X}_{\mathrm{s}} \bigcap \mathcal{X}_{\mathrm{t}}$ to form bridge feature set $\mathcal{X}_{\Lambda}$.

(2) **For** $\mathcal{X}_{\Lambda_i}$ in $\mathcal{X}_{\Lambda}$
- construct pseudo labeled datasets: $\mathcal{D}_{\mathrm{s}}^{\Lambda_i}$ and $\mathcal{D}_{\mathrm{t}}^{\Lambda_i}$
- train linear classifiers on them: $f_{\mathcal{D}_{\mathrm{s}}^{\Lambda_i}}$ and $f_{\mathcal{D}_{\mathrm{t}}^{\Lambda_i}}$
- add $\boldsymbol{\theta}_{\Lambda_i}^{(\mathrm{s})}$ into $\boldsymbol{\Theta}_{\Lambda}^{(\mathrm{s})}$, $\boldsymbol{\theta}_{\Lambda_i}^{(\mathrm{t})}$ into $\boldsymbol{\Theta}_{\Lambda}^{(\mathrm{t})}$

End

(3) Run SCL procedure to learn a single-view correspondence: $\boldsymbol{\Theta}_{\Lambda}$.

(4) Run MVPCA procedure: $\boldsymbol{\mathcal{P}}_{\Lambda} = \mathrm{MVPCA}([\boldsymbol{\Theta}_{\Lambda}^{(\mathrm{s})}, \boldsymbol{\Theta}_{\Lambda}^{(\mathrm{t})}, \boldsymbol{\Theta}_{\Lambda}])$.

(5) $\widehat{\mathcal{D}}_{\mathrm{sl}} = \mathit{FeatureExpansion}(\mathcal{D}_{\mathrm{sl}}, \boldsymbol{P}_{\Lambda})$
$\widehat{\mathcal{D}}_{\mathrm{t}} = \mathit{FeatureExpansion}(\mathcal{D}_{\mathrm{t}}, \boldsymbol{\mathcal{P}}_{\Lambda})$

(6) Train a linear classifier $\widehat{f}_{\mathcal{D}_{\mathrm{sl}}}$ on $\widehat{\mathcal{D}}_{\mathrm{sl}}$.
$\widehat{f}_{\mathcal{D}_{\mathrm{sl}}}$ is the output.

4 Experimental Result Analysis

4.1 Data and Software Specification

In this paper, we perform experiments on both multi-domain sentiment dataset① and 20 Newsgroups dataset②, which show that MVPCA/VMVPCA can not only outperform baseline non-transfer method substantially, but also improve the performance of transfer learning method SCL significantly.

We choose Multi-Domain Sentiment Dataset to show that polysemous words do exist in sentiment domain. While polysemous words incorporate conflict correspondence between words into *single-view* algorithm SCL, experiments show MVPCA/VMVPCA can solve this problem and further improve the performance of transfer learning. Multi-Domain Sentiment Dataset is crawled from Amazon, where there are four types of product reviews: Book, DVD, Electronic and Kitchen. Every product review is given a rating from 1 to 5. Reviews with rating higher than 3 are regarded as positive instances, and reviews with rating lower than 3 are labeled as negative with others discarded. After this conversion, every domain contains 1000 positive instances and 1000 negative instances. Besides these labeled instances, every domain has some unlabeled instances. Book domain has 6000 unlabeled instances, DVD has 34 741, Electronic has 13 153, and Kitchen domain has 16 785 unlabeled instances. Details are shown in Table 4.

Before applying MVPCA and VMVPCA to Multi-Domain dataset, we briefly outline the distributional shift between domains by showing different distributions of top 50 bridge features in Book, DVD, Electronic and Kitchen domains in Fig.1. The X axis denotes different bridge features in alphabetic order. The Y axis denotes the value of mutual information. From Fig.1, we can see that all of the four domains differ substantially in the mutual information values of bridge features. In this way, Fig.1 does show some intuitive differences between domains, although the differences can be detected in more ways than mutual information. These distributional differences encourage us to apply transfer learning techniques to cross domain sentiment classification.

Table 4. Multi-Domain Sentiment Dataset Specification

Domain	Positive	Negative	Unlabeled	Features
Book	1 000	1 000	6 000	42 835
DVD	1 000	1 000	34 741	162 682
Electronic	1 000	1 000	13 153	47 426
Kitchen	1 000	1 000	16 785	48 534

①http://www.cs.jhu.edu/∼mdredze/datasets/sentiment/.

②http://people.csail.mit.edu/jrennie/20Newsgroups/.

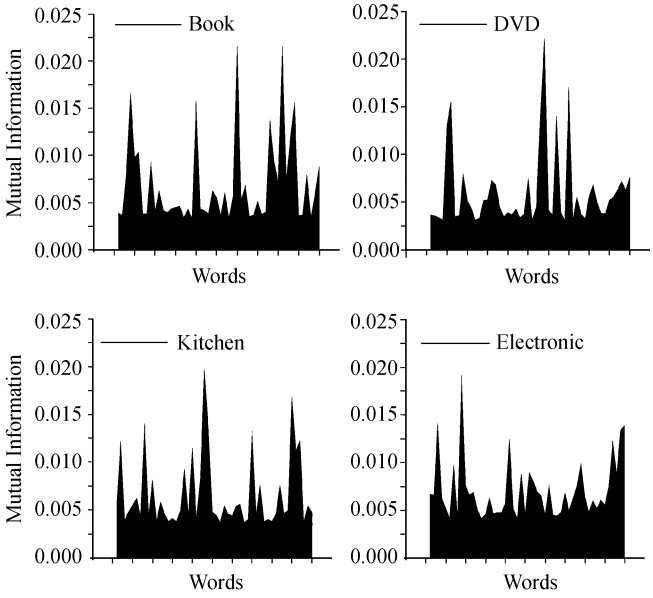

Fig.1. Distributional shift in different domains.

We choose 20 Newsgroups dataset to show that polysemous words exist not only in sentiment review data, but also in topic domain text data. The 20 Newsgroups dataset is a set of nearly 20 000 documents collected from 20 different newsgroups. These 20 different newsgroups covers extensive topics from computer and science to forsale. These topics can be categorized into 6 top categories and 20 subcategories, with each subcategory corresponding to a newsgroup. They are shown in Table 5. In order to make the 20 Newsgroups dataset

Table 5. 20 Newsgroups Dataset Specification

comp.graphics	rec.autos
comp.os.ms-windows.misc	rec.motorcycles
comp.sys.ibm.pc.hardware	rec.sport.baseball
comp.sys.mac.hardware	rec.sport.hockey
comp.windows.x	
sci.crypt	talk.politics.misc
sci.electronics	talk.politics.guns
sci.med	talk.politics.mideast
sci.space	
misc.forsale	talk.religion.misc
	alt.atheism
	soc.religion.christian

Table 6. Cross Domain Classification Setting of the 20 Newsgroups Dataset

Dataset	Source Domain	Target Domain
comp vs. sci	comp.graphics	comp.sys.ibm.pc.hardware
(C-S)	comp.os.ms-windows.misc	comp.sys.mac.hardware
	sci.crypt	comp.windows.x
	sci.electronics	sci.med
		sci.space
rec vs. talk	rec.autos	rec.sport.baseball
(R-T)	rec.motorcycles	rec.sport.hockey
	talk.politics.guns	talk.politics.mideast
	talk.politics.misc	talk.religion.misc
rec vs. sci	rec.autos	rec.motorcycles
(R-S)	rec.sport.baseball	rec.sport.hockey
	sci.med	sci.crypt
	sci.space	sci.electronics
sci vs. talk	sci.electronics	sci.crypt
(S-T)	sci.med	sci.space
	talk.politics.misc	talk.politics.guns
	talk.religion.misc	talk.politics.mideast
comp vs. rec	comp.graphics	comp.os.ms-windows.misc
(C-R)	comp.sys.ibm.pc.hardware	comp.windows.x
	comp.sys.mac.hardware	rec.autos
	rec.motorcycles	rec.sport.baseball
	rec.sport.hockey	
comp vs. talk	comp.graphics	comp.os.ms-windows.misc
(C-T)	comp.sys.mac.hardware	comp.sys.ibm.pc.hardware
	comp.windows.x	talk.politics.guns
	talk.politics.mideast	talk.politics.misc
	talk.religion.misc	

suitable for our problem setting, we reorganize the 20 subcategories and put them in related but different domains. In this paper, the settings of 20 Newsgroups dataset is similar to Dai *et al.*[7] We reorganize the 20 subcategories into 6 source and target domain pairs. Within each domain pair, the texts are from only two top categories. And within each domain in the pair, positive instances consist of some subcategories in one top category while negative instances consist of some other subcategories in the other top category. Details of cross domain classification setting of the 20 Newsgroups dataset are shown in Table 6.

4.2 Experiments on Multi-Domain Sentiment Dataset

In order to conduct the comparisons fairly, we compare MVPCA/VMVPCA and SCL in the same experimental settings: 1) MVPCA/VMVPCA and SCL use the same parameter settings, e.g., the same number of bridge features, m, and the same dimensionality of latent semantic space, p; 2) MVPCA/VMVPCA and SCL utilize the same linear classifier when learning feature correspondences: logistic regression. Then we show the experimental results according to different parameter settings in Table 7 and Table 8.

In Table 7 and Table 8, "B-D" in "Domain" column means that source domain is Book domain and target domain is DVD domain. "Baseline" column denotes the accuracy of baseline method, which first trains a logistic regression model on source domain, and then evaluates the testing accuracy on target domain instances. m denotes the number of bridge features. p denotes the number of eigenvectors generated by MVPCA or SCL[1-2], and it is also the dimensionality of cross domain semantic space. "S" denotes the accuracy of SCL[1-2]; "M" denotes the accuracy of MVPCA; "VM" denotes the accuracy of the VMVPCA algorithm described in Table 3, regarded as a mixture algorithm of SCL and MVPCA.

From Table 7 and Table 8, it is clear that MVPCA and VMVPCA win most of the time over SCL algorithm and baseline method. The values of accuracy in bold mean they are the winners, and the values in bold

Table 7. Classification Accuracy Comparison on Multi-Domain Sentiment Dataset (%)

Domain	Baseline	$m = 100$, $p = 10$			$m = 100$, $p = 30$			$m = 100$, $p = 50$			$m = 100$, $p = 70$		
		S	M	VM	S	M	VM	S	M	VM	S	M	VM
B-D	79.30	78.45	77.05	77.85	79.15	78.90	79.05	79.80	80.60	81.05	80.65	80.70	**80.90**
B-E	77.20	76.25	76.55	76.50	79.30	78.80	77.95	79.55	78.90	79.95	79.05	78.60	**80.00**
B-K	77.15	77.40	76.10	75.90	79.20	78.75	79.25	80.00	80.05	79.85	**80.70**	79.25	**80.70**
D-B	75.10	76.45	74.95	74.60	78.50	77.75	77.90	78.25	**78.80**	78.30	78.60	78.55	78.35
D-E	76.50	77.60	76.20	76.40	76.90	78.25	78.15	77.20	78.25	**79.90**	77.60	78.70	79.15
D-K	76.75	76.95	77.50	77.05	80.25	78.50	78.75	80.30	78.50	79.65	**80.85**	79.15	79.80
E-B	70.30	72.75	70.45	71.90	72.15	72.45	72.80	71.80	71.75	**73.30**	71.75	71.85	72.70
E-D	71.55	71.20	69.45	71.45	73.55	72.80	71.85	72.75	**73.80**	73.30	72.75	75.50	73.10
E-K	84.75	85.35	85.45	85.55	85.75	86.95	86.85	86.55	87.65	**87.95**	86.85	87.60	87.50
K-B	71.70	72.85	72.50	73.05	73.85	73.30	72.05	**74.70**	72.75	73.30	74.35	72.55	74.30
K-D	72.05	73.30	73.30	71.80	75.85	74.40	75.25	74.10	**76.45**	75.85	75.00	75.80	76.20
K-E	84.25	84.50	85.10	84.30	85.45	85.60	85.70	86.15	86.15	**86.55**	85.55	85.50	86.05

Table 8. Classification Accuracy Comparison on Multi-Domain Sentiment Dataset (%)

Domain	Baseline	$m = 200$, $p = 10$			$m = 200$, $p = 30$			$m = 200$, $p = 50$			$m = 200$, $p = 70$		
		S	M	VM	S	M	VM	S	M	VM	S	M	VM
B-D	79.30	79.95	79.05	78.75	79.20	79.55	79.65	80.05	81.00	80.30	**81.80**	81.65	81.75
B-E	77.20	76.40	76.95	76.70	77.70	77.30	77.30	79.35	78.60	79.05	78.95	79.35	**80.25**
B-K	77.15	76.75	76.80	76.60	78.05	78.50	78.25	81.25	80.00	80.10	81.40	80.15	**81.45**
D-B	75.10	77.05	75.95	74.90	77.30	78.60	78.40	76.40	77.15	76.65	**77.90**	77.25	**77.90**
D-E	76.50	77.50	76.20	76.20	73.30	77.65	79.10	75.20	77.65	78.70	77.15	78.30	**79.55**
D-K	76.75	77.60	76.95	77.50	80.20	78.45	78.35	79.35	77.60	77.65	**80.95**	78.35	80.35
E-B	70.30	72.45	70.45	70.85	72.35	**72.55**	72.30	72.10	71.30	71.95	71.30	70.25	72.50
E-D	71.55	71.30	69.20	69.25	73.55	72.55	72.20	71.60	74.20	73.15	73.20	**76.65**	75.35
E-K	84.75	85.70	85.70	85.45	85.35	86.80	86.25	85.95	87.10	86.55	87.00	**87.60**	87.35
K-B	71.70	71.95	72.45	73.05	73.05	71.90	72.05	73.70	72.20	73.15	**74.70**	72.05	73.20
K-D	72.05	72.60	72.15	71.25	71.20	74.10	75.70	73.90	74.75	75.20	73.30	75.25	**75.20**
K-E	84.25	84.60	83.05	83.30	83.65	85.60	85.55	85.25	85.40	85.60	85.70	**85.85**	85.65

with underline show the winners are MVPCA or VMVPCA. To see how they perform in different settings, we need to conduct further analysis.

Firstly, we put some efforts on the win/draw/loss statistics of these methods under different settings of p shown in Table 9, where "M vs. Baseline" denotes the comparison between MVPCA and baseline method. Similarly for VM vs. Baseline. "M-best vs. S-best" denotes the comparison between the best results of MVPCA/VMVPCA and the best results of SCL, under different settings of p. "M/VM vs. S" compares the better results of MVPCA and VMVPCA with SCL. "M vs. S" denotes the comparison between MVPCA and SCL. Similarly for "VM vs. S", "VM vs. M". Obviously, both MVPCA and VMVPCA outperform baseline method in almost all time of transfer learning when $p \geqslant 30$. When the parameter $p = 10$ or 30, MVPCA and VMVPCA do not outperform SCL or perform even worse when $p = 10$. When $p > 30$, performances of MVPCA and VMVPCA are enhanced and they outperform SCL significantly. And they achieve the most significant improvement over SCL when $p = 50$. When $m = 100$, $p \geqslant 50$, VMVPCA performs better than SCL and baseline at the 5% significance level, while MVPCA performs comparable with SCL and significant better than baseline; when $m = 200$, $p \geqslant 30$, VMVPCA performs better than SCL and baseline at the 5% significance level, while MVPCA performs comparable with SCL and significant better than baseline. However, it is notable that when p reaches 70, MVPCA degenerates its performance to be comparable with SCL. In general, when p reaches 50, MVPCA and VMVPCA can obviously improve SCL's performance. The reason why they need a certain number of eigenvectors to get better knowledge transfer performance is that they performs a multi-view component analysis which deals with more complex structural information across domains than SCL does. So compared with SCL, MVPCA and VMVPCA need more eigenvectors to depict the knowledge transfer clearly.

Table 9. Win/Draw/Loss Statistics Analysis on Multi-Domain Sentiment Dataset

Parameter Settings	$p = 10$	$p = 30$	$p = 50$	$p = 70$
M vs. Baseline	12/0/12	23/0/1	24/0/0	23/0/1
VM vs. Baseline	9/0/15	23/0/1	24/0/0	24/0/0
M-best vs. S-best	17/2/5	17/2/5	17/2/5	17/2/5
M/VM vs. S	9/0/15	13/0/11	17/0/7	18/0/6
M vs. S	7/2/15	11/0/13	14/1/9	12/0/12
VM vs. S	7/0/17	12/0/12	16/0/8	17/0/7
VM vs. M	11/1/12	11/1/12	16/0/8	16/0/8

Fig.2 and Fig.3 demonstrate the similar statistics of the comparison between SCL and MVPCA/VMVPCA. X axis denotes the settings of source and target domain pairs. Y axis denotes the classification accuracy. When $m = 100$, $p = 10$ and $m = 200$, $p = 10$, it is shown that MVPCA/VMVPCA only achieve comparable performance with baseline method. And at the same time, SCL performs better than our algorithms. When $p = 30$ and $m = 200$, SCL degenerates its performance, and both MVPCA and VMVPCA achieve obvious improvement over SCL and baseline method.

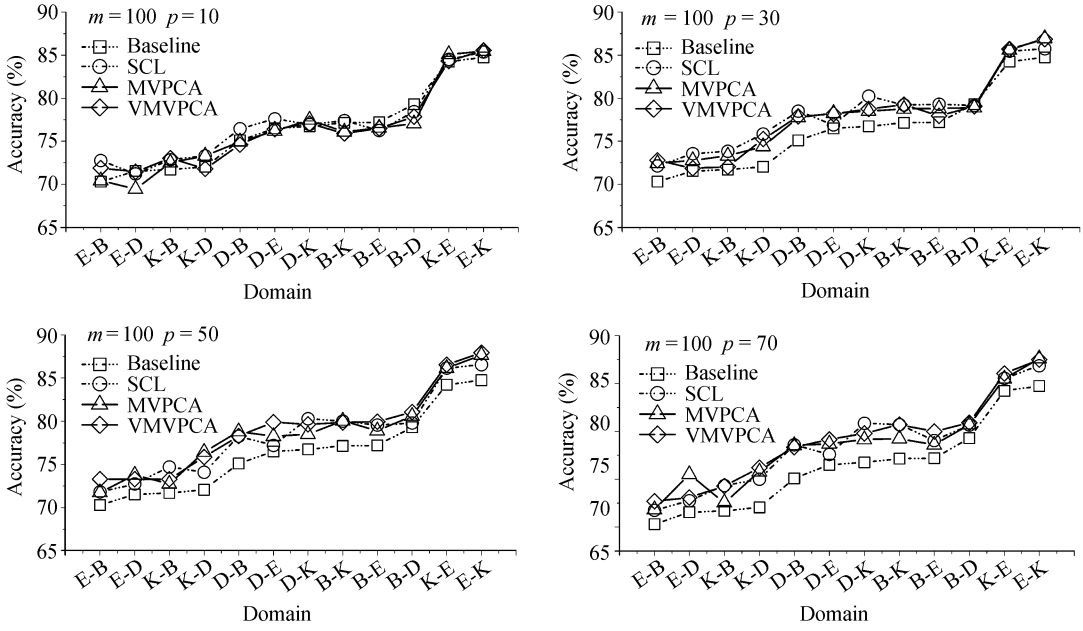

Fig.2. Transfer learning performance comparison on multi-domain sentiment dataset when $m = 100$.

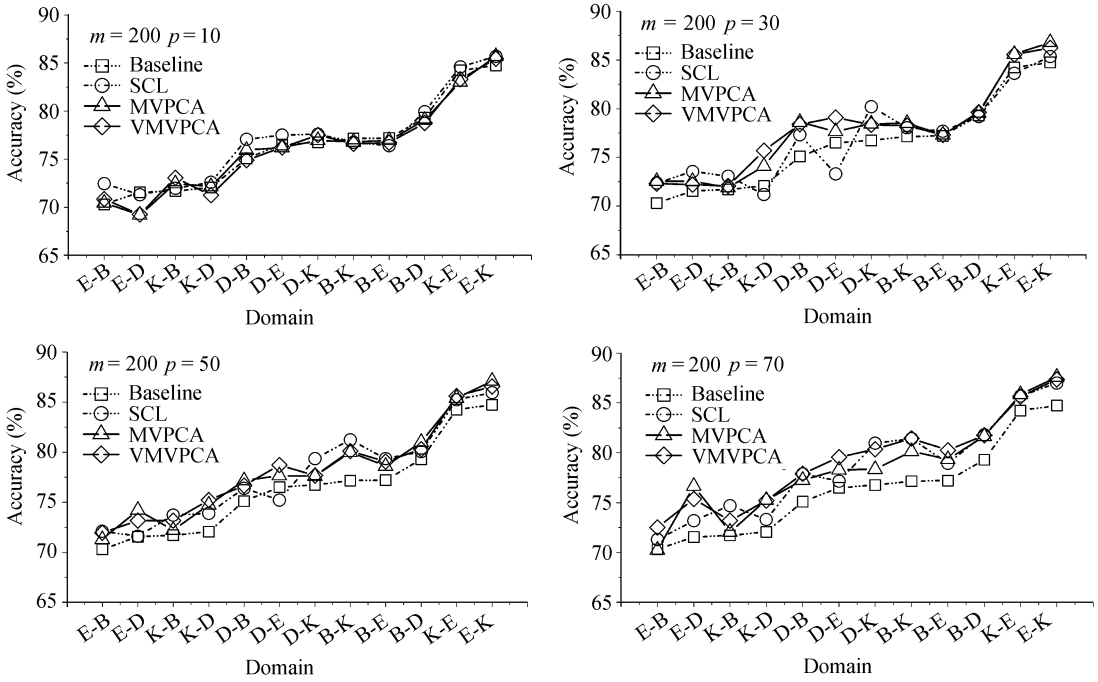

Fig.3. Transfer learning performance comparison on Multi-Domain Sentiment Dataset when $m = 200$.

When $p = 30$ and $m = 100$, both SCL and MVPCA/VMVPCA get enhanced performance, with SCL achieving the best performance. When $p = 50$ and $p = 70$, the performance curves of MVPCA/VMVPCA stay over the cure of SCL most of the time. And MVPCA/VMVPCA win 35 in 48 times of comparison when $p = 50$ and $p = 70$. In general, MVPCA and VMVPCA achieve significant improvement over SCL when p reaches 50 or 70.

Then we study how the performances of MVPCA/VMVPCA are affected by parameters p and m. Table 10 shows the average classification accuracy of the algorithms in comparison under different settings of p. If we focus on one line, when p increases, the cross domain classification accuracy of every algorithm will increase. This trend makes sense, because if one classifier owns more information to benefit knowledge transfer, we can expect the continuous improvement. From Fig.4, we can also see that at first when p is small, SCL performs the best. As soon as MVPCA collects enough eigenvectors to facilitate transfer learning, MVPCA outperforms SCL. One more remark about the parameter m is that as m increases, we cannot always expect to improve its performance. Because choose more bridge features cannot guarantee that the discriminative power increases at the same time, they will always increase the information. Bridge features within one domain have so similar explanations that 100 bridge features can give sufficient helpful information for transfer learning. This also leaves an open question for future work on choosing bridge features.

From above discussions, we get the point that MVPCA/VMVPCA need a certain number of eigenvectors to facilitate cross domain knowledge transfer, and when p reaches 50, MVPCA/VMVPCA achieve significant improvement over SCL and baseline method. So let us take a detailed look at the improvement of our algorithms. Fig.5 shows error reduction ratios of MVPCA/VMVPCA over baseline method and SCL algorithm. X axis denotes source and target domain pairs; Y axis denotes error reduction ratio, and the curves of MVPCA/VMVPCA are computed with the minimum error of MVPCA and VMVPCA. In this paragraph, we simply denote MVPCA and VMVPCA

Table 10. Average Performance Comparison on Multi-Domain Sentiment Dataset (%)

	$m = 100$				$m = 200$			
	$p = 10$	$p = 30$	$p = 50$	$p = 70$	$p = 10$	$p = 30$	$p = 50$	$p = 70$
Baseline	76.38				76.38			
SCL	**76.92**	**78.33**	78.43	78.64	**76.99**	77.08	77.84	78.61
MVPCA	76.22	78.04	78.64	78.65	76.24	77.80	78.08	78.56
VMVPCA	76.36	77.96	**79.08**	**79.06**	76.15	**77.93**	**78.17**	**79.21**

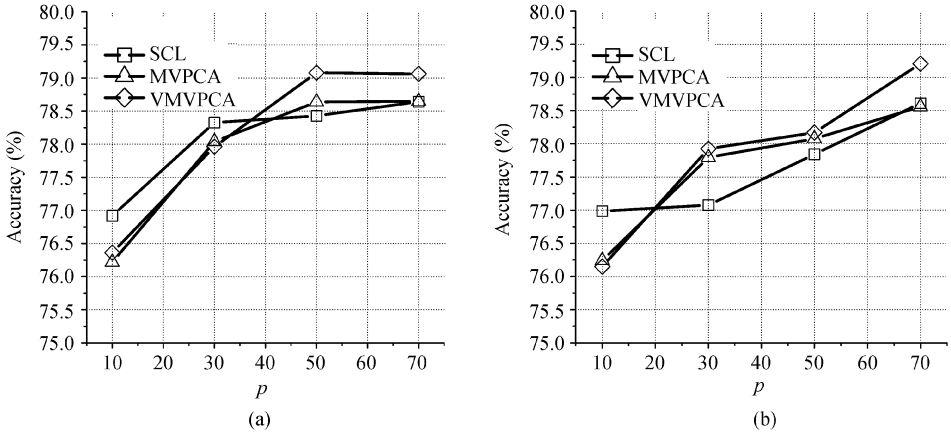

Fig.4. Average transfer learning performance comparison on multi-domain sentiment dataset. (a) $m = 100$. (b) $m = 200$.

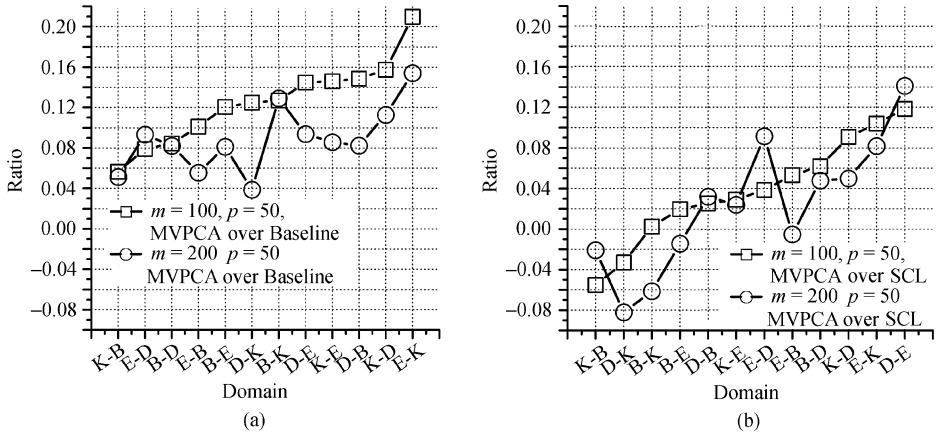

Fig.5. Error reduction ratio of MVPCA on multi-domain sentiment dataset.

together by MVPCA. Compared with baseline method, when $p = 50$, MVPCA always gets a positive error reduction, which means MVPCA always outperforms baseline method under this setting. MVPCA reduces the error of baseline method transferring from Electronic to Kitchen up to 21% when $m = 100$ and $p = 50$, and achieves an average of 11.43% error reduction over baseline when $m = 100$, $p = 50$. Compared with SCL algorithm, it seems that transferring from Kitchen domain to Book domain and transferring from DVD domain to Kitchen domain are two difficult tasks for MVPCA/VMVPCA, and they cannot outperform SCL algorithm in many different parameter settings. However, MVPCA can reduce SCL's error of transferring from DVD to Electronic up to 14% when $m = 200$, $p = 50$, and MVPCA achieves an average of 2.7% error reduction over SCL when $m = 100$, $p = 50$.

Finally, we can give the following remarks of our algorithm. When p increases to a certain extend, MVPCA collects enough information from the complex structure of *multi-view* feature correspondences, and these information keeps MVPCA away from incorrect correlations. Thus, MVPCA can outperform the *single-view* transfer learning algorithm SCL. It is more interesting that when we fuse the *single-view* correspondences with the *multi-view* correspondences, we can learn a more powerful semantic space to benefit our knowledge transfer. Thus VMVPCA achieves an average error reduction of 11.43% over baseline method and 2.7% over SCL algorithm.

4.3 Experiments on 20 Newsgroups Dataset

Similar to the experiments on multi-domain sentiment dataset, we perform MVPCA/VMVPCA and SCL under the setting of the same parameters and the same base classifier: logistic regression. The experimental results of comparisons between MVPCA/VMVPCA and SCL are listed in Table 11, where "Domain" tells the setting of source and target domains, and "C-S" in this column denotes that source and target domains are from dataset "comp vs. sci" in Table 5. "Baseline", "S", "M" and "VM" are defined similarly with Table 7. On 20 Newsgroups dataset, we

Table 11. Classification Accuracy Comparison on 20 Newsgroups Dataset (%)

Domain	Baseline	$m = 100$, $p = 10$			$m = 100$, $p = 30$			$m = 100$, $p = 50$			$m = 100$, $p = 70$		
		S	M	VM	S	M	VM	S	M	VM	S	M	VM
C-S	68.51	68.74	68.90	72.03	66.00	70.99	72.09	65.86	70.48	**71.81**	66.20	**71.78**	70.23
C-R	81.95	88.40	86.24	87.84	**89.78**	85.47	88.73	88.55	86.92	88.17	88.12	86.11	87.33
C-T	87.03	90.95	90.59	90.65	91.50	93.32	**93.85**	91.20	93.32	93.68	88.61	91.72	93.16
R-S	72.65	75.30	76.48	78.75	76.23	77.39	78.68	76.31	78.98	**79.38**	76.81	78.15	77.82
R-T	74.29	66.58	59.80	72.77	67.23	59.66	75.41	62.28	62.87	**76.01**	66.47	62.64	75.78
S-T	75.41	73.89	70.00	73.63	74.26	68.24	74.81	75.10	69.68	75.49	**75.80**	70.81	74.99

only perform experiments when $m = 100$ to show the superior of MVPCA/VMVPCA over SCL.

We need to conduct further analysis to see how they perform in different settings. Table 12 gives the win/draw/loss statistics of MVPCA/VMVPCA compared with other methods. When we conduct t-test, VMVPCA is significantly better than baseline method and SCL, while MVPCA is significantly better than baseline method and comparable with SCL. Fig.6 shows the comparisons between MVPCA/VMVPCA and other methods intuitively. X axis denotes different source and target domain pairs. Y axis denotes the prediction accuracy. Fig.6 shows that VMVPCA almost always stays above SCL and baseline. It also shows that R-T and S-T are two hard domains for transfer learning. While SCL performs much worse than baseline on domain R-T, which is called negative transfer, VMVPCA performs better than both SCL and baseline when $p > 10$.

Table 13 and Fig.7 give the overview of how these methods perform on 20 Newsgroups dataset. The numbers in bold with underline in Table 13 denote the best performances from MVPCA/VMVPCA. In Fig.7, it can be seen that the performance curves are different from those on multi-domain sentiment dataset.

Table 12. Win/Draw/Loss Statistics Analysis on 20 Newsgroups Dataset

Parameter Settings	$p = 10$	$p = 30$	$p = 50$	$p = 70$
M vs. Baseline	4/0/2	4/0/2	4/0/2	4/0/2
VM vs. Baseline	4/0/2	5/0/1	6/0/0	5/0/1
(M/VM)-best vs. S-best	4/0/2	4/0/2	4/0/2	4/0/2
M/VM vs. S	3/0/3	5/0/1	5/0/1	4/0/2
M vs. S	2/0/4	3/0/3	4/0/2	3/0/3
VM vs. S	3/0/3	5/0/1	5/0/1	5/0/1
VM vs. M	6/0/0	6/0/0	6/0/0	4/0/2

Table 13. Average Performance Comparison on 20 Newsgroups Dataset (%)

$m = 100$	$p = 10$	$p = 30$	$p = 50$	$p = 70$
Baseline	76.64	76.64	76.64	76.64
SCL	77.31	77.50	76.55	77.00
MVPCA	75.34	75.85	77.04	76.87
VMVPCA	**79.28**	**80.60**	**80.76**	**79.89**

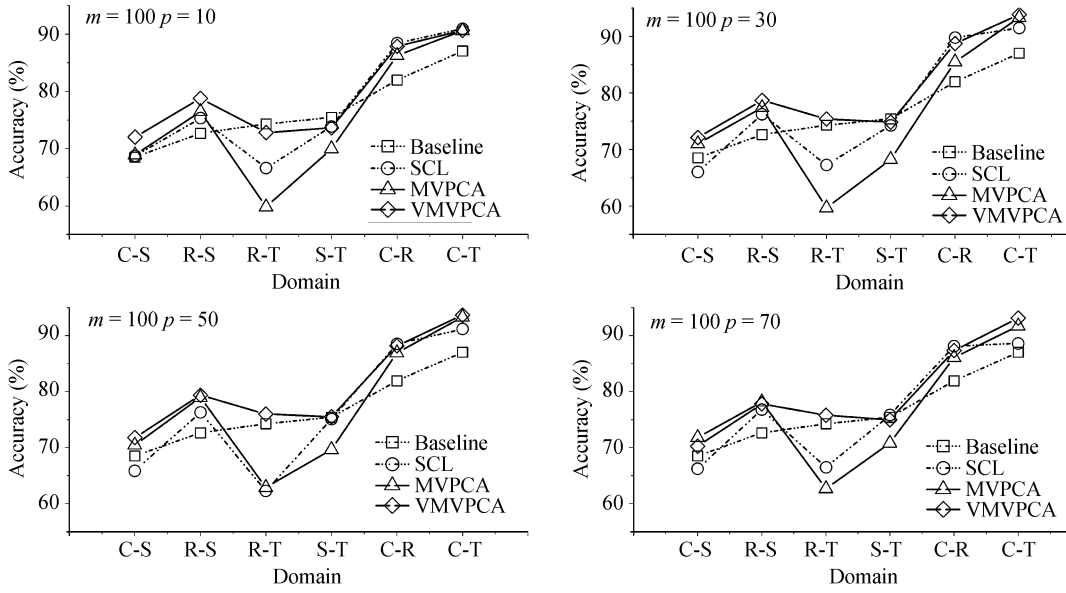

Fig.6. Transfer learning performance comparison between MVPCA/VMVPCA and other methods on 20 Newsgroups dataset.

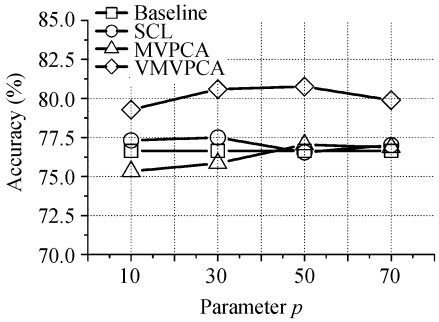

Fig.7. Average transfer learning performance comparison between baseline method, SCL algorithm, and MVPCA/VMVPCA on 20 Newsgroups dataset.

The curves in Fig.7 are rather stable compared to those in Fig.4, with VMVPCA staying higher above the curves of baseline and SCL methods. However, MVPCA needs more eigenvectors to achieve its best performance than SCL method. And this phenomenon is the same as that MVPCA shown on Multi-Domain Sentiment Dataset. Fig.8 shows the error reduction of MVPCA/VMVPCA over SCL algorithm. When $p = 50$, VMVPCA reduces the prediction error of SCL up to 22% on domain R-T. On 20 Newsgroups dataset, we can conclude that VMVPCA performs significantly better than SCL with MVPCA achieving comparable performance with SCL, while all of MVPCA/VMVPCA perform substantially better than baseline method. The improvement of MVPCA/VMVPCA over *single-view* transfer learning algorithm SCL and baseline non-transfer method shows that *multi-view* transfer learning technique works well in topic domain text data.

4.4 Experiments on the Performance of MVPCA/VMVPCA under Different Quantities of Polysemous Words

To show MVPCA/VMVPCA outperforms SCL actually due to the problem of polysemous words, we further study the performance of MVPCA/VMVPCA and SCL under different quantities of vague words in the dataset. We specify a polysemous word by its correlation with class labels in different domains, and to change the number of polysemous words in a domain, we artificially create some polysemous words to incorporate into the dataset. Thus the perturbed datasets provide different settings of quantities of vague words to study how the performance is affected by the number of polysemous words. Experiments show that the more polysemous words in the dataset, MVPCA/VMVPCA the better choice MVPCA/VMVPCA is when we need transfer learning techniques.

To further study the performance of MVPCA/VMVPCA under different quantities of polysemous words, we randomly choose "B-E", "E-K" from multi-domain sentiment dataset, and "C-S", "R-T" from 20 Newsgroups dataset. To specify whether a word w is a polysemous word, we simply compute

$$p(w = 1, y = +1) = \frac{count(w = 1, y = +1)}{\sum_{w,y} count(w, y)}$$

and

$$p(w = 1, y = -1) = \frac{count(w = 1, y = -1)}{\sum_{w,y} count(w, y)}.$$

If $\frac{p(w=1,y=+1)}{p(w=1,y=-1)} > \delta$ with $\delta \geqslant 1$, it tells that w occurs more often with positive label. If $\frac{p(w=1,y=+1)}{p(w=1,y=-1)} < \frac{1}{\delta}$ with $\delta \geqslant 1$, it tells that w occurs more often with negative label. In this paper, we set $\delta = 1$. So if $\frac{p(w=1,y=+1)}{p(w=1,y=-1)} > 1$ occurs in source domain and meanwhile $\frac{p(w=1,y=+1)}{p(w=1,y=-1)} < 1$ occurs in target domain, or vice versa, we regard w as a polysemous word with different meanings in different domains. Based on the definition of polysemous words, we can artificially create some new polysemous words. For example, if a word "cars"

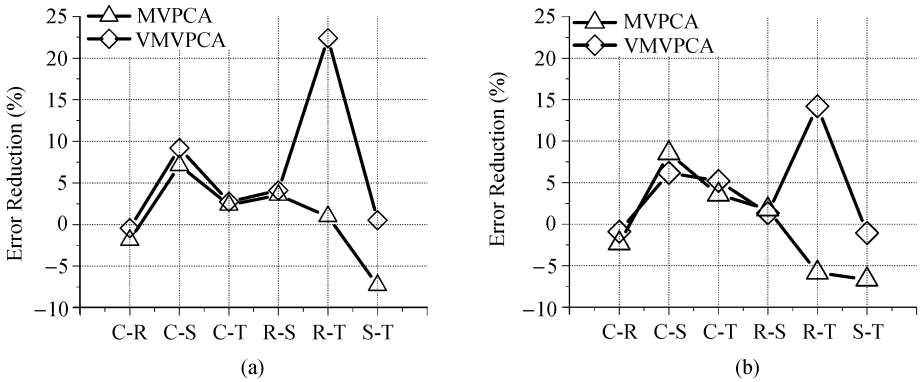

Fig.8. Error reduction of MVPCA/VMVPCA over SCL on 20 Newsgroups dataset. X axis denotes different domain pairs, Y axis denotes the percentage of prediction error reduction of MVPCA/VMVPCA over SCL. (a) $m = 100, p = 50$. (b) $m = 100, p = 70$.

has strong correlation with positive label in source domain, while "israeli" strongly correlates with negative label in target domain, we generate "cars_israeli" as a polysemous word. Similarly, if "fbi" correlates more with negative label in source domain while "games" highly correlates with positive label in target domain, "fbi_games" is artificially generated as a polysemous word. Table 14 illustrates some of the artificial polysemous words in different domains.

After the generation of new words, we replace those original words with these artificially generated polysemous words so that we can perturb the dataset. In the experiments, we perturb the datasets with artificially generating 40, 80, 120, 160 and 200 new vague words respectively. After the perturbation, the occurrence ratio of polysemous words in the dataset varies depending on the number of artificial polysemous words. Table 15 shows the different polysemous words ratios before and after perturbation.

When we perform MVPCA/VMVPCA and other algorithms on the perturbed datasets, we get the results in Table 16 and Fig.9. In Table 16, "B-E", "E-K", "C-S", "R-T" denote different sub-datasets. "AW" denotes different settings of artificially generated polysemous words. "Baseline", "S", "M", "VM" denote the same results as defined in Table 7. m and p are algorithm parameters. Under different settings of m and p, we compare the cross domain prediction accuracy between

Table 14. Artificially Generated Polysemous Words in Different Datasets

B-E	excellent_not work, a must_not, wonderful_not buy, great_return, highly recommend_waste, i highly_terrible, an excellent_waste your, easy_work, a great_refund, a wonderful_after, easy_poor,
E-K	great_not, excellent_waste, price_not buy, highly_disappointed, a great_was, easy to_returned, the price_return, the best_after, perfect_your money,
C-S	windows_space, graphics_earth, dos_orbit, file_years, files_nasa, thanks_writes, microsoft_medical, image_shuttle, gif_article, win_disease
R-T	bike_jews, car_government, cars_israeli, engine_arab, ride_turkish, motorcycle_war, honda_by, bmw_people, riding_children, bikes_such

Table 15. Polysemous Words Ratios with Perturbation in Different Domains

No. Artificial Words	B-E	E-K	C-S	R-T
0	0.26	0.19	0.45	0.27
40	0.29	0.22	0.49	0.31
80	0.31	0.24	0.52	0.35
120	0.33	0.27	0.56	0.37
160	0.35	0.28	0.58	0.40
200	0.37	0.30	0.61	0.44

Table 16. Experiments on the Performance of MVPCA/VMVPCA under Different Quantities of Polysemous Words

	AW	Base-	$m = 100$, $p = 10$			$m = 100$, $p = 30$			$m = 100$, $p = 50$			$m = 100$, $p = 70$		
		line	S	M	VM	S	M	VM	S	M	VM	S	M	VM
B-E	40	71.25	69.15	64.50	**69.20**	**72.25**	67.50	70.05	71.65	70.35	**71.70**	72.00	70.10	**72.35**
	80	67.75	**67.65**	63.45	66.70	**68.60**	65.50	67.75	**68.50**	65.85	68.15	**68.40**	66.90	67.95
	120	66.20	**66.20**	60.80	65.75	**66.95**	63.60	66.40	**67.45**	63.70	66.85	**67.30**	64.10	66.80
	160	65.35	**66.40**	61.45	64.75	65.85	61.20	**65.95**	**66.25**	62.75	65.35	64.95	63.15	**65.50**
	200	63.30	**63.35**	60.35	63.10	**64.05**	59.85	63.90	62.25	59.10	**63.45**	**61.65**	59.10	61.60
E-K	40	81.90	82.60	82.15	**82.95**	82.70	81.75	**82.90**	82.75	82.30	**82.90**	82.55	81.95	**82.90**
	80	82.10	81.95	81.75	**82.05**	81.15	81.20	**82.15**	**82.10**	80.70	82.00	81.60	81.40	**82.15**
	120	80.80	**80.45**	**80.45**	**80.45**	80.75	80.20	**81.55**	81.20	80.40	**81.55**	80.15	**81.20**	**81.20**
	160	80.25	**80.25**	79.60	80.00	79.05	79.80	**80.90**	79.75	**80.35**	**80.35**	80.00	79.75	**80.55**
	200	80.05	79.35	79.45	**79.60**	79.50	79.15	**80.45**	79.65	79.15	**80.10**	**79.90**	79.45	79.75
C-S	40	61.21	61.87	61.89	**63.16**	58.39	**63.14**	62.99	58.66	**62.69**	62.63	59.01	**61.99**	61.64
	80	59.80	59.60	59.27	**61.03**	55.76	**60.15**	59.37	57.06	58.68	**59.21**	**59.13**	56.76	58.31
	120	58.60	59.62	59.97	**60.66**	57.19	**59.87**	58.84	56.04	58.66	**58.78**	55.82	58.19	**58.82**
	160	58.23	**60.42**	59.27	59.87	56.27	57.43	**59.48**	55.57	57.98	**59.19**	55.67	58.37	**59.82**
	200	59.25	**60.95**	59.37	60.48	57.31	56.78	**59.15**	57.64	57.35	**61.52**	56.51	58.80	**61.50**
R-T	40	48.80	46.05	42.42	**48.64**	37.38	41.55	**46.61**	39.63	42.67	**45.34**	40.93	43.57	**46.67**
	80	44.30	**45.65**	41.55	45.51	35.95	39.72	**44.44**	36.51	40.00	**46.24**	32.60	40.23	**47.06**
	120	40.14	39.78	40.73	**42.90**	35.78	37.58	**38.54**	34.18	38.34	**40.51**	34.21	37.78	**42.78**
	160	41.24	44.02	40.90	**46.41**	36.06	39.63	**41.60**	36.60	38.48	**44.36**	35.67	40.00	**42.48**
	200	39.41	40.25	41.21	**43.94**	37.72	40.73	**43.07**	35.95	39.83	**42.64**	36.34	39.41	**42.03**

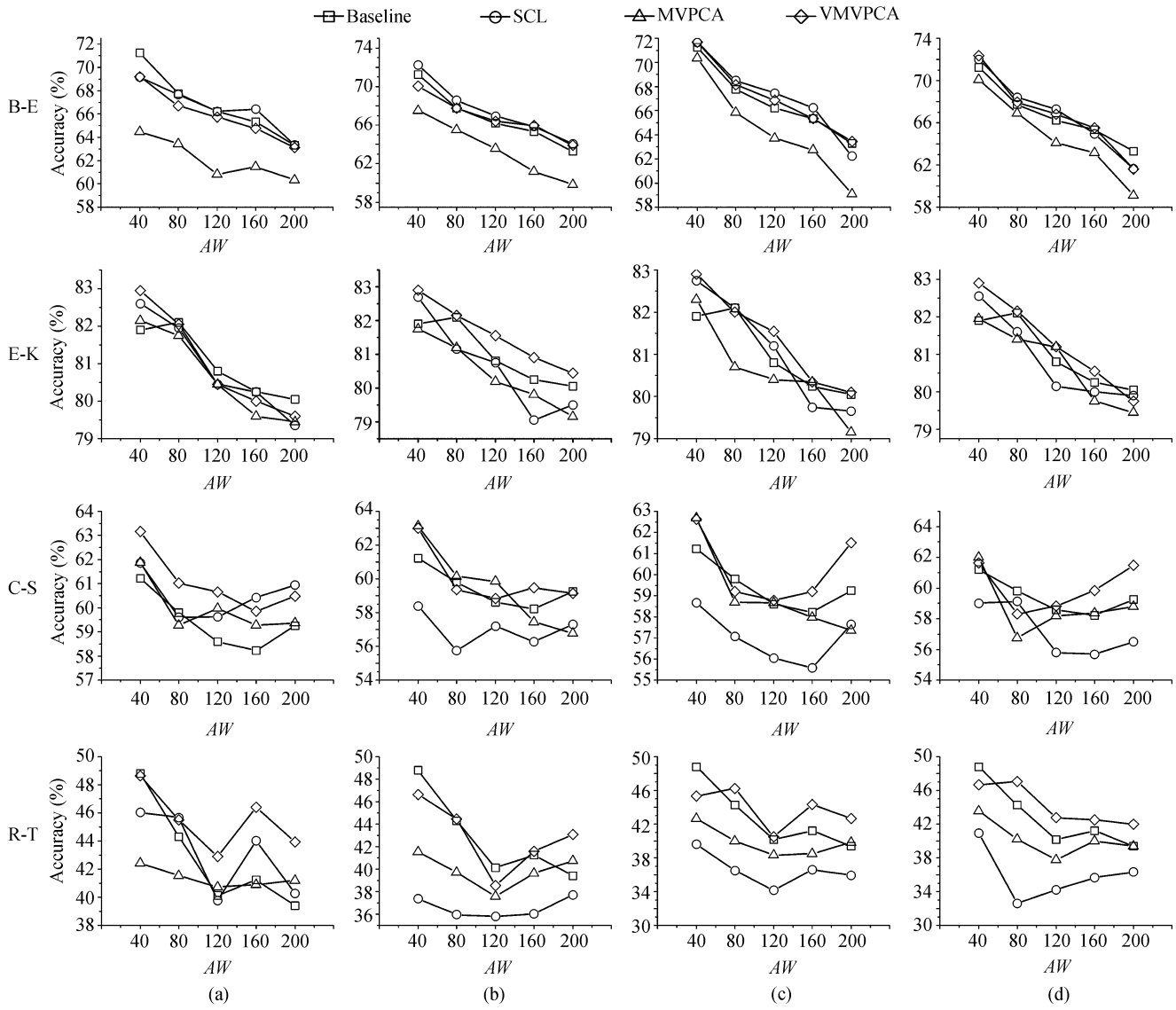

Fig.9. Transfer learning performance vs. quantity of artificial polysemous words. (Every row corresponds to a sub-dataset, which is labeled to the left of it. In each row, every subgraph illustrates the performances of all the algorithms under a fixed parameter setting. In every subgraph, X axis denotes the number of artificial generated vague words, Y axis denotes the transfer prediction accuracy on the perturbed dataset.) (a) $m = 100$, $p = 10$. (b) $m = 100$, $p = 30$. (c) $m = 100$, $p = 50$. (d) $m = 100$, $p = 70$.

SCL and MVPCA/VMVPCA. Numbers in bold with underline show that MVPCA/VMVPCA are winners; numbers in bold show that SCL wins. Table 16 illustrates that VMVPCA achieves comparable performance with SCL in the sub-dataset of B-E, while MVPCA is hurt by the artificially generated polysemous words performing worse than baseline and SCL. In the sub-dataset of E-K, while MVPCA performs comparable with SCL, VMVPCA performs better than SCL at the 5% significance level. In the sub-datasets of C-S and R-T, MVPCA/VMVPCA perform much better than SCL under different settings of m, p and quantity of artificial polysemous words. When $m = 100$, $p = 70$ and $AW = 80$, VMVPCA improves the prediction accuracy of SCL up to 44.56% on R-T sub-dataset.

One important advantage of VMVPCA over SCL is that VMVPCA rarely perform worse than baseline method when $p \geqslant 30$ shown in Fig.9. Another advantage of VMVPCA is that the performance of SCL decreases much faster than VMVPCA, when the number of polysemous words increases. To understand why VMVPCA does not improve SCL significantly on sub-dataset B-E due to the artificial polysemous words, we may need to refer to Blitzer *et al.*[2], who proposes a distance measure between domains, which is called $\mathcal{A}$-*distance*. Measured by $\mathcal{A}$-*distance*, Book and Electronic

domains have so large a distance that incorporating 200 new polysemous words into B-E sub-dataset does not change it much. Thus, the advantage of VMVPCA over SCL is not significant. Another point needs to be noted is that the improvement of VMVPCA over SCL on C-S and R-T sub-datasets is much higher than on B-E and E-K sub-datasets. This difference is caused by the different polysemous word ratios of each sub-dataset. In Table 15, C-S and R-T sub-datasets have much higher polysemous words ratios than B-E and E-K sub-datasets. With higher polysemous words ratio, we can get more improvement of VMVPCA over SCL. Based on above discussions, we recommend the *multi-view* transfer learning algorithm VMVPCA compared with SCL, especially when polysemous words occur a lot in source and target domains.

5 Conclusion and Future Work

This paper proposes a new method called *Multi-View Principal Component Analysis* for transfer learning. MVPCA/VMVPCA separates the *single-view* of correspondences into several parts, which can be regarded as *multi-view* correspondences. In these separated correspondences, contradict correlations are eliminated, and then the cross domain correlation could be extracted by *multi-view* PCA. In this way, MVPCA/VMVPCA learns a neater and cleaner semantic space to help knowledge transfer.

However, in order to learn the semantic space of *multi-view* correspondences, Canonical Correlation Analysis[20-21] is a natural way to solve the problem. In our future work, we will study Canonical Correlation Analysis in depth, and apply it to transfer learning. What is more, we can embed the discriminative information to learn a cross domain semantic space[22-24]. Besides, considering text summarization[25], Canonical Correlation Analysis could also be helpful. And there are some open questions, for example, how to choose a set of bridge features that are of high quality.

Acknowledgements We thank Yue Lu, Yu-Feng Li, Xiang-Nan Kong, Jun-Ming Xu, Yin Zhu, and Cheng-Xiang Zhai for helpful comments on this draft. We also thank anonymous reviewers for great comments on improving the work and presentation of this paper, and editors for their hard work.

References

[1] Blitzer J, McDonald R, Pereira F. Domain adaptation with structural correspondence learning. In *Proc. Conference on Empirical Methods in Natural Language*, Sydney, Australia, Jul. 22-23, 2006, pp.120-128.

[2] Blitzer J, Dredze M, Pereira F. Biographies, Bollywood, Boomboxes and Blenders: Domain adaptation for sentiment classification. In *Proc. the 45th Annual Meeting of the Association of Computational Linguistics*, Prague, Czech, Jun. 25-27, 2007, pp.432-439.

[3] Dhillon I S, Mallela S, Modha D S. Information-theoretic co-clustering. In *Proc. the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (*KDD*), Washington DC, USA, Aug. 24-27, 2003, pp.89-98.

[4] Pan S J, Tsang I W, Kwok J T, Yang Q. Domain adaptation via transfer component analysis. In *Proc. the 21st International Joint Conference on Artificial Intelligence*, Pasadena, USA, Jul. 11-17, 2009, pp.1187-1192.

[5] Daumé H. Frustratingly easy domain adaptation. In *Proc. the 45th Annual Meeting of the Association for Computational Linguistics*, Prague, Czech, Jun. 25-27, 2007, pp.256-263.

[6] Daumé H, Marcu D. Domain adaptation for statistical classifiers. *Journal of Artificial Intelligence Research*, 2006, 26: 101-126.

[7] Dai W Y, Xue G R, Yang Q, Yu Y. Co-clustering based classification for out-of-domain documents. In *Proc. the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Jose, USA, Aug. 12-15, 2007, pp.210-219.

[8] Pan S J, Kwok J T, Yang Q, Pan J J. Adaptive localization in a dynamic WiFi environment through multi-view learning. In *Proc. the 22nd AAAI Conference on Artificial Intelligence*, Vancouver, Canada, Jul. 22-26, 2007, pp.1108-1113.

[9] Dai W Y, Yang Q, Xue G R, Yu Y. Boosting for transfer learning. In *Proc. the 24th International Conference on Machine Learning*, Corvalis, USA, Jun. 20-24, 2007, pp.193-200.

[10] Huang J, Smola A, Gretton A, Borgwardt K M, Schölkopf B. Correcting sample selection bias by unlabeled data. In *Proc. the 19th Annual Conference on Neural Information Processing Systems*, Vancouver, Canada, Dec. 4-6, 2006, pp.601-608.

[11] Zadrozny B. Learning and evaluating classifiers under sample selection bias. In *Proc. the 21st International Conference on Machine Learning*, Banff, Canada, Jul. 4-8, 2004, pp.903-910.

[12] Jiang J, Zhai C X. Instance weighting for domain adaptation in NLP. In *Proc. the 45th Annual Meeting of the Association of Computational Linguistics*, Prague, Czech, Jun. 25-27, 2007, pp.264-271.

[13] Raina R, Battle A, Lee H, Packer B, Ng A Y. Self-taught learning: Transfer learning from unlabeled data. In *Proc. the 24th International Conference on Machine Learning*, Corvalis, USA, Jun. 20-24, 2007, pp.759-766.

[14] Lee H, Battle A, Raina R, Ng A Y. Efficient sparse coding algorithms. In *Proc. the 19th Annual Conference on Neural Information Processing Systems*, Vancouver, Canada, Dec. 4-6, 2006, pp.801-808.

[15] Tan S, Cheng X, Wang Y, Xu H. Adapting naive Bayes to domain adaptation for sentiment analysis. In *Proc. the 31st European Conference on IR Research*, Toulouse, France, Apr. 7-9, 2009, pp.337-349.

[16] Dai W Y, Xue G R, Yang Q, Yu Y. Transferring Naive Bayes classifiers for text classification. In *Proc. the 22nd AAAI Conference on Artificial Intelligence* (*AAAI 2007*), Vancouver, Canada, Jul. 22-26, 2007, pp.540-545.

[17] Sandler T, Blitzer J, Talukdar P, Pereira F. Regularized learning with networks of features. In *Proc. the 22nd Annual Conference on Neural Information Processing Systems*, Vancouver, Canada, Dec. 8-11, 2008, pp.1401-1408.

[18] Vinokourov A, Shawe-Taylor J, Cristianini N. Inferring a semantic representation of text via cross-language correlation analysis. In *Proc. the 15th Advances of Neural Information Processing Systems*, Vancouver, Canada, Dec. 9-12, 2002, pp.1473-1480.

[19] Li Y, Shawe-Taylor J. Using KCCA for Japanese-English

cross-language information retrieval and classification. *Journal of Intelligent Information Systems*, 2006, 27(2): 117-133.
[20] Hardoon D R, Szedmak S, Shawe-Taylor J. Canonical correlation analysis: An overview with application to learning methods, *Neural Computation*, 2004, 16(12): 2639-2664.
[21] Hotelling H. Relations between two sets of variates. *Biometrika*, 1936, 28(3/4): 321-377.
[22] Diethe T, Hardoon D R, Shawe-Taylor J. Multiview Fisher discriminant analysis. In *Proc. Learning from Multiple Sources Workshop* (*NIPS 2008*), Whistler, Canada, Dec. 13, 2008.
[23] Zhou Z H, Zhan D C, Yang Q. Semi-supervised learning with very few labeled training examples. In *Proc. the 22nd AAAI Conference on Artificial Intelligence* (*AAAI 2007*), Vancouver, Canada, Jul. 22-26, 2007, pp.675-680.
[24] Zhang D Q, Zhou Z H, Chen S C. Semi-supervised dimensionality reduction. In *Proc. the 7th SIAM International Conference on Data Mining* (*SDM 2007*), Minneapolis, USA, Apr. 26-28, 2007, pp.629-634.
[25] Long C, Huang M L, Zhu X Y, Ming Li. A new approach for multi-document update summarization. *Journal of Computer Science and Technology*, 2010, 25(4): 739-749.

Yang-Sheng Ji received his Bachelor's degree in computer science from Nanjing University, China, in 2006 and he is now pursuing a Ph.D. degree in Department of Computer Science and Technology, Nanjing University. He visited Machine Learning Group in Microsoft Research Asia as an intern in 2008. From November 2009 to November 2010, he visited University of Illinois at Urbana-Champaign, and worked with Prof. Cheng-Xiang Zhai. His research interests include sentiment analysis, review summarization, and transfer learning.

Jia-Jun Chen is a professor of the Department of Computer Science & Technology at Nanjing University, China. He received his Ph.D. degree in computer science from Nanjing University, China, in 1998. Currently, Prof. Chen is the director of the Natural Language Processing Lab at Nanjing University. He is a member of China Computer Federation (CCF), and a member of the Machine Translation Specialization Committee of Chinese Information Processing Society of China. His research interests include machine translation, text categorization and information extraction. His research has been supported by the National 863 High-Tech Projects of China, the National Natural Science Foundation of China (NSFC), the Social Science Foundation of China and the Natural Science Foundation of Jiangsu Province of China.

Gang Niu received his Master's degree from Department of Computer Science and Technology in Nanjing University, China. He is currently a Ph.D. candidate in Department of Computer Science in Tokyo Institute of Technology, Japan. His research interests include support vector machines, spectral clustering and manifold learning algorithms.

Lin Shang is an associate professor in the Department of Computer Science and Technology at the Nanjing University. She received her B.S. and M.S. degrees from Lanzhou University, China, in 1995 and 1998, and Ph.D. degree in computer science from the Nanjing University in 2004. She has visited Regina University, Canada, in May, 2007. Her research interests mainly include artificial intelligence, machine learning, data mining, and rough sets. In these areas she has published over 50 papers in the past five years. She is the program committee member of RSKT'07, RSKT'08, CRSSC 2006, 2007, 2008, 2009, and organization committee chair/co-chair of agent2008, CRSSC 2010, China. She is a member of CCF.

Xin-Yu Dai received his Ph.D. degree in computer science from Nanjing University in 2005, China. He is currently an associate professor of the Department of Computer Science and Technology at Nanjing University. His research interests focus on machine translation and information retrieval. He is a member of CCF.