Cross-Domain Sentiment Classification via Spectral Feature Alignment*

Sinno Jialin Pan†, Xiaochuan Ni‡, Jian-Tao Sun‡, Qiang Yang† and Zheng Chen‡
†Department of Computer Science and Engineering
Hong Kong University of Science and Technology, Hong Kong
‡Microsoft Research Asia, Beijing, P. R. China
†{sinnopan, qyang}@cse.ust.hk, ‡{xini, jtsun, zhengc}@microsoft.com

ABSTRACT

Sentiment classification aims to automatically predict sentiment polarity (e.g., positive or negative) of users publishing sentiment data (e.g., reviews, blogs). Although traditional classification algorithms can be used to train sentiment classifiers from manually labeled text data, the labeling work can be time-consuming and expensive. Meanwhile, users often use some different words when they express sentiment in different domains. If we directly apply a classifier trained in one domain to other domains, the performance will be very low due to the differences between these domains. In this work, we develop a general solution to sentiment classification when we do not have any labels in a target domain but have some labeled data in a different domain, regarded as source domain. In this cross-domain sentiment classification setting, to bridge the gap between the domains, we propose a spectral feature alignment (**SFA**) algorithm to *align* domain-specific words from different domains into unified clusters, with the help of domain-independent words as a bridge. In this way, the clusters can be used to reduce the gap between domain-specific words of the two domains, which can be used to train sentiment classifiers in the target domain accurately. Compared to previous approaches, **SFA** can discover a robust representation for cross-domain data by fully exploiting the relationship between the domain-specific and domain-independent words via simultaneously co-clustering them in a common latent space. We perform extensive experiments on two real world datasets, and demonstrate that **SFA** significantly outperforms previous approaches to cross-domain sentiment classification.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: [Text Mining]; I.2.6 [**Artificial Intelligence**]: [Learning]

General Terms

Algorithms, Experimentation

Keywords

Domain Adaptation, Sentiment Classification, Feature Alignment, Transfer Learning, Opinion Mining

*This work was done when the first author was on an internship at Microsoft Research Asia.

WWW 2010, April 26–30, 2010, Raleigh, North Carolina, USA.
ACM 978-1-60558-799-8/10/04.

1. INTRODUCTION

With the explosion of Web 2.0 services, more and more user-generated sentiment data have been shared on the Web. They exist in the form of user reviews on shopping or opinion sites, in posts of blogs or customer feedback. As a result, opinion mining has attracted much attention recently [29, 24], for example, opinion summarization [19, 26], opinion integration [25] and review spam identification [21], etc. Sentiment classification, which aims at classifying sentiment data into polarity categories (e.g., positive or negative), is widely studied because many users do not explicitly indicate their sentiment polarity thus we need to predict it from the text data generated by users.

In literature, supervised learning algorithms [30] have been proved promising and widely used in sentiment classification. However, the performance of these methods relies on manually labeled training data. In some cases, the labeling work may be time-consuming and expensive in order to build accurate sentiment classifiers. Furthermore, these approaches are domain dependent. The reason is that users may use domain-specific words to express sentiment in different domains. Table 1 shows several user review sentences from two domains: *electronics* and *video games*. In the *electronics* domain, we may use words like "compact", "sharp" to express our positive sentiment and use "blurry" to express our negative sentiment. While in the *video game* domain, words like "hooked", "realistic" indicate positive opinion and the word "boring" indicates negative opinion. Due to the mismatch between domain-specific words, a sentiment classifier trained in one domain may not work well when directly applied to other domains. Thus cross-domain sentiment classification algorithms are highly desirable to reduce domain dependency and manually labeling cost.

In this paper, we target at finding an effective approach for the cross-domain sentiment classification problem. Assume we have a set of labeled data from a source domain, in order to train a classifier for a target domain, we leverage some unlabeled data from the target domain to help. In detail, we propose a *spectral feature alignment* (**SFA**) algorithm to find a new representation for cross-domain sentiment data, such that the gap between domains can be reduced. **SFA** uses some *domain-independent words* as a bridge to construct a bipartite graph to model the co-occurrence relationship between *domain-specific words* and *domain-independent words*. The idea is that if two domain-specific words have connections to more common domain-independent words in the graph, they tend to be aligned together with higher probability. Similarly, if two domain-independent words have connections to more common domain-specific words in the graph, they tend to be aligned together with higher probability. We adapt a spectral clustering algorithm, which is based on the graph spectral theory [9], on the bipartite graph to co-align domain-specific and domain-independent

Table 1: Cross-domain sentiment classification examples: reviews of *electronics* and *video games* products. Boldfaces are domain-specific words, which are much more frequent in one domain than in the other one. Italic words are some domain-independent words, which occur frequently in both domains. "+" denotes positive sentiment, and "-" denotes negative sentiment.

	electronics	*video games*
+	**Compact**; easy to operate; very *good* picture quality; looks **sharp**!	A very *good* game! It is action packed and full of *excitement*. I am very much **hooked** on this game.
+	I purchased this unit from Circuit City and I was very *excited* about the quality of the picture. It is really *nice* and **sharp**.	Very **realistic** shooting action and *good* plots. We played this and were **hooked**.
-	It is also quite **blurry** in very dark settings. I will *never buy* HP again.	The game is so **boring**. I am extremely unhappy and will probably *never buy* UbiSoft again.

words into a set of feature-clusters. In this way, the clusters can be used to reduce the mismatch between domain-specific words of both domains. Finally, we represent all data examples with these clusters and train sentiment classifiers based on the new representation. Different from state-of-the-art cross-domain sentiment classification algorithms such as the structural correspondence learning (**SCL**) algorithm [6], our proposed **SFA** can fully exploit the relationship between domain-independent and domain-specific words via co-aligning them on the bipartite graph to learn a more compact and meaningful representation underlying the graph. Experiments in two real world domains indicate that **SFA** is indeed promising in obtaining better performance than several baselines including **SCL** [6] in terms of the accuracy for cross-domain sentiment classification.

The rest of the paper is organized as follows. In the next section, we first describe the problem we study and give some definitions. Then we present the idea behind our proposed feature alignment approach in Section 3. The details of our solution are presented in Section 4. We conduct a series of experiments to evaluate the effectiveness of our proposed solution in Section 5. Finally, we review some related works in Section 6 and conclude our work in Section 7.

2. PROBLEM SETTING

Before giving a formal definition of the problem we address in this paper, we first present some definitions.

Definition 1 (**Domain**) A domain D denotes a class of entities in the world or a semantic concept.

For example, different types of products, such as books, dvds and furniture, can be regarded as different domains. Take research area as another example, computer science, mathematics and physics can be also regarded as different domains.

Definition 2 (**Sentiment**) Given a specific domain D, sentiment data are the text documents containing user opinions about entities of the domain. User sentiment may exist in the form of a sentence, paragraph or article. In either case, it corresponds with a sequence of words $w_1 w_2 ... w_{x_j}$, where w_i is a word from a vocabulary W. In this work, we represent user sentiment data with a bag-of-words method, with $c(w_i, x_j)$ to denote the frequency of word w_i in x_j, and the word sequential information is ignored.

Without loss of generality, we use a unified vocabulary W for all domains and $|W| = m$. Furthermore, in sentiment classification tasks, either single word or NGram can be used as features to represent sentiment data, thus in the rest of this paper, we will use *word* and *feature* interchangeably.

Definition 3 (**Labeled / Unlabeled Sentiment Data**) Given a specific domain D, the sentiment data x_i and a y_i denoting the polarity of x_i, x_i is said to be *positive* if the overall sentiment expressed in x_i is positive ($y_i = +1$), while x_i is *negative* if the overall sentiment expressed in x_i is negative ($y_i = -1$). A pair of sentiment text and its corresponding sentiment polarity $\{x_i, y_i\}$ is called the *labeled sentiment data*. If x_i has no polarity assigned, it is *unlabeled sentiment data*.

Besides positive and negative sentiment, there are also neutral and mixed sentiment data in practical applications. *Mixed polarity* means user sentiment is positive in some aspects but negative in other ones. *Neutral polarity* means that there is no sentiment expressed by users. In this paper, we only focus on positive and negative sentiment data, but it is not hard to extend the proposed solution to address multi-category sentiment classification problems.

Based on the definitions described above, we now define the problem we try to address in this paper as follows;

Problem Definition (**Cross-domain Sentiment Classification**) Given two specific domains D_{src} and D_{tar}, where D_{src} and D_{tar} are referred to as a source domain and a target domain respectively, suppose we have a set of labeled sentiment data $\mathcal{D}_{src} = \{(x_{src_i}, y_{src_i})\}_{i=1}^{n_{src}}$ in D_{src}, and some unlabeled sentiment data $\mathcal{D}_{tar} = \{x_{tar_j}\}_{j=1}^{n_{tar}}$ in D_{tar}. The task of cross-domain sentiment classification is to learn an accurate classifier to predict the polarity of unseen sentiment data from D_{tar}.[1]

In order to solve this problem, we propose a framework which targets to achieve two subtasks: (1) To identify domain-independent features and (2) to align domain-specific features. In the first subtask, we aims to learn a feature selection function $\phi_{DI}(\cdot)$ to select l domain-independent features, which occur frequently and act similarly across domains D_{src} and D_{tar}. These domain-independent features are used as a bridge to make knowledge transfer across domains possible. After identifying domain-independent features, we can use $\phi_{DS}(\cdot)$ to denote a feature selection function for selecting domain-specific features, which can be defined as the complement of domain-independent features. In the second subtask, we aims to to learn an alignment function $\varphi : \mathbb{R}^{(m-l)} \rightarrow \mathbb{R}^k$ to align domain-specific features from both domains into k predefined feature clusters $z_1, z_2, ..., z_k$, s.t. the difference between domain specific features from different domains on the new representation constructed by the learned clusters can be dramatically reduced.

For simplicity, we use W_{DI} and W_{DS} to denote the vocabulary of domain-independent and domain-specific features respectively. Then sentiment data x_i can be divided into two disjoint views. One view consists of features in W_{DI}, and the other is composed of features in W_{DS}. We use $\phi_{DI}(x_i)$ and $\phi_{DS}(x_i)$ to denote the two views respectively.

[1]Note that, in this paper we only consider one source domain and one target domain. However, our proposed method is quite general and can be easily adapted to solve multi-source domain adaptation problems.

3. A MOTIVATING EXAMPLE

In this section, we use an example to introduce the motivation of our solution to the cross-domain sentiment classification problem. First of all, we assume the sentiment classifier f is a linear function, which can be written as

$$y^* = f(x) = \mathbf{sgn}(xw^T),$$

where $x \in \mathbb{R}^{1\times m}$ and $\mathbf{sgn}(xw^T) = +1$ if $xw^T \geq 0$, otherwise, $\mathbf{sgn}(xw^T) = -1$. w is the weight vector of the classifier, which can be learned from a set of training data (pairs of sentiment data and their corresponding polarity labels).

Consider the example shown in Table 1 to illustrate our idea. We use a standard bag-of-words method to represent sentiment data of the *electronics* (**E**) and *video games* (**V**) domains. From Table 2, we can see that the difference between domains is caused by the frequency of the domain-specific words. Domain-specific words in the **E** domain, such as *compact*, *sharp*, *blurry*, do not occur in the **V** domain. On the other hand, domain-specific words in the **V** domain, such as *hooked*, *realistic*, *boring*, do not occur in the **E** domain. Suppose the **E** domain is the source domain and the **V** domain is the target domain, our goal is to train a vector of weights w^* with labeled data from the **E** domain, and use it to predict sentiment polarity for the **V** domain data.[2] Based on the three training sentences in the **E** domain, the weights of features such as *compact* and *sharp* should be positive. The weight of features such as *blurry* should be negative and the weights of features such as *hooked*, *realistic* and *boring* can be arbitrary or zeros if a L_1 regularizer is applied on w for model training. However, an ideal weight vector in the **V** domain should have positive weights for features such as *hooked*, *realistic* and a negative weight for the feature *boring*, while the weights of features such as *compact*, *sharp* and *blurry* may take arbitrary values. That is why the classifier learned from the **E** domain may not work well in the **V** domain.

Table 2: Bag-of-words representations of *electronics* (E) and *video games* (V) reviews. Only domain-specific features are considered. "..." denotes all other words.

		...	compact	sharp	blurry	hooked	realistic	boring
E	+	...	1	1	0	0	0	0
	+	...	0	1	0	0	0	0
	-	...	0	0	1	0	0	0
V	+	...	0	0	0	1	0	0
	+	...	0	0	0	1	1	0
	-	...	0	0	0	0	0	1

In order to reduce the mismatch between features of the source and target domains, a straightforward solution is to make them more similar by adopting a new representation. Table 3 shows an ideal representation of domain-specific features. Here, *sharp_hooked* denotes a cluster consisting of *sharp* and *hooked*, *compact_realistic* denotes a cluster consisting of *compact* and *realistic*, and *blurry_boring* denotes a cluster consisting of *blurry* and *boring*. We can use these clusters as high-level features to represent domain-specific words. Based on the new representation, the weight vector w^* trained in the **E** domain should be also an ideal weight vector in the **V** domain. In this way, based on the new representation, the classifier learned from one domain can be easily adapted to another one.

The problem is how to construct such an ideal representation as shown in Table 3. Clearly, if we directly apply traditional clustering algorithms such as k-means [18] on Table 2, we are not able to

[2]For simplicity, we only discuss domain-specific words here and ignore all other words.

Table 3: Ideal representations of domain-specific words.

		...	sharp_hooked	compact_realistic	blurry_boring
E	+	...	1	1	0
	+	...	1	0	0
	-	...	0	0	1
V	+	...	1	0	0
	+	...	1	1	0
	-	...	0	0	1

align *sharp* and *hooked* into one cluster, since the *distance* between them is large. In order to reduce the gap and align domain-specific words from different domains, we can utilize domain-independent words as a bridge. As shown in Table 1, words such as *sharp*, *hooked*, *compact* and *realistic* often co-occur with other words such as *good* and *exciting*, while words such as *blurry* and *boring* often co-occur with a word *never_buy*. Since the words like *good*, *exciting* and *never_buy* occur frequently in both the **E** and **V** domains, they can be treated as domain-independent features. Table 4 shows co-occurrences between domain-independent and domain-specific words. It is easy to find that, by applying clustering algorithms such as k-means on Table 4, we can get the feature clusters shown in Table 3: *sharp_hooked*, *blurry_boring* and *compact_realistic*.

Table 4: A co-occurrence matrix of domain-specific and domain-independent words.

	compact	realistic	sharp	hooked	blurry	boring
good	1	1	1	1	0	0
exciting	0	0	1	1	0	0
never_buy	0	0	0	0	1	1

So, the co-occurrence relationship between domain-specific and domain-independent features is useful for feature alignment across different domains. In this paper, we use a bipartite graph to represent this relationship and then adapt spectral clustering techniques to find a new representation for domain-specific features. In the following section, we will present spectral domain-specific feature alignment algorithm in detail.

4. SPECTRAL DOMAIN-SPECIFIC FEATURE ALIGNMENT

In this section, we describe our algorithm to adapt spectral clustering techniques to align domain-specific features from different domains for cross-domain sentiment classification.

4.1 Domain-Independent Feature Selection

First of all, we need to identify which features are domain independent. As mentioned above, domain-independent features should occur frequently and act similarly in both the source and target domains. In this section, we present several strategies for selecting domain-independent features.

A first strategy is to select domain-independent features based on their frequency in both domains. More specifically, given the number l of domain-independent features to be selected, we choose features that occur more than k times in both the source and target domains. k is set to be the largest number such that we can get at least l such features.

A second strategy is based on the mutual dependence between features and labels on the source domain data. In [6], mutual information is applied on source domain labeled data to select features as "pivots", which can be referred to as domain-independent features in this papers. In information theory, mutual information is

used to measure the mutual dependence between two random variables. Feature selection using mutual information can help identify features relevant to source domain labels. But there is no guarantee that the selected features act similarly in both domains.

Here we propose a third strategy for selecting domain-independent features. Motivated by the supervised feature selection criteria, we can use mutual information to measure the dependence between features and domains. If a feature has high mutual information, then it is domain specific. Otherwise, it is domain independent. Furthermore, we require domain-independent features occur frequently. So, we modify the mutual information criterion between features and domains as follows,

$$I(X^i; D) = \sum_{d \in D} \sum_{x \in X^i, x \neq 0} p(x, d) log_2 \left(\frac{p(x, d)}{p(x)p(d)} \right), \quad (1)$$

where D is a domain variable and we only sum over non-zero values of a specific feature X^i. The smaller $I(X^i; D)$ is, the more likely that X^i can be treated as a domain-independent feature.

4.2 Bipartite Feature Graph Construction

Based on the above strategies for selecting domain-independent features, we can identify which features are domain independent and which ones are domain specific. Given domain-independent and domain-specific features, we can construct a bipartite graph $G = (V_{DS} \bigcup V_{DI}, E)$ between them. In G, each vertex in V_{DS} corresponds to a domain-specific word in W_{DS}, and each vertex in V_{DI} corresponds to a domain-independent word in W_{DI}. An edge in E connects two vertexes in V_{DS} and V_{DI} respectively. Note that there is no intra-set edges linking two vertexes in V_{DS} or V_{DI}. Furthermore, each edge $e_{ij} \in E$ is associated with a non-negative weight m_{ij}. The score of m_{ij} measures the relationship between word $w_i \in W_{DS}$ and $w_j \in W_{DI}$ in $\mathcal{D}_{src}$ and $\mathcal{D}_{tar}$ (e.g., the total number of co-occurrence of $w_i \in W_{DS}$ and $w_j \in W_{DI}$ in $\mathcal{D}_{src}$ and $\mathcal{D}_{tar}$). A bipartite graph example is shown in Figure 1, which is constructed based on the example shown in Table 4. So we can use the constructed bipartite graph to model the intrinsic relationship between domain-specific and domain-independent features.

Besides using the co-occurrence frequency of words within documents, we can also adopt more meaningful methods to estimate m_{ij}. For example, we can define a reasonable "window size". If a domain-specific word and a domain-independent word co-occur within the "window size", then there is an edge connecting them. Furthermore, we can also use the distance between w_i and w_j to adjust the score of m_{ij}. The smaller is their distance, the larger weight we can assign to the corresponding edge. In this paper, for simplicity, we set the "window size" to be the maximum length of all documents. Also we do not consider word position to determine the weights for edges. We want to show that by constructing a simple bipartite graph and adapting spectral clustering techniques on it, we can algin domain-specific features effectively.

4.3 Spectral Feature Clustering

In the previous section, we have presented how to construct a bipartite graph between domain-specific and domain-independent features. In this section, we show how to adapt a spectral clustering algorithm on the feature bipartite graph to align domain-specific features.

In graph spectral theory [9], there are two main assumptions: (1) if two nodes in a graph are connected to many common nodes, then they should be very similar (or quite related), (2) there is a low-dimensional latent space underlying a complex graph, where two nodes are similar to each other if they are similar in the original graph. Based on these two assumptions, spectral graph theory has been widely applied in many problems, e.g., dimensionality reduction and clustering [27, 3, 14]. In our case, we assume (1) if two domain-specific features are connected to many common domain-independent features, then they tend to be very related and will be aligned to a same cluster with high probability, (2) if two domain-independent features are connected to many common domain-specific features, then they tend to be very related and will be aligned to a same cluster with high probability, (3) we can find a more compact and meaningful representation for domain-specific features, which can reduce the gap between domains. Therefore, with the above assumptions, we expect the mismatch problem between domain-specific features can be alleviated by applying graph spectral techniques on the feature bipartite graph to discover a new representation for domain-specific features.

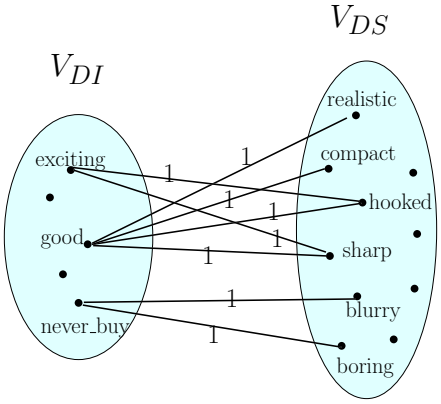


Figure 1: A bipartite graph example of domain-specific and domain-independent features based on Table 4.

Before we present how to adapt a spectral clustering algorithm to align domain-specific features, we first briefly introduce a standard spectral clustering algorithm [27] as follows,

Given a set of points $V = \{v_1, v_2, ..., v_n\}$ and their corresponding weighted graph G, the goal is to cluster the points into k clusters, where k is an input parameter.

1. Form an affinity matrix for V: $\mathbf{A} \in \mathbb{R}^{n \times n}$, where $\mathbf{A}_{ij} = m_{ij}$, if $i \neq j$; $\mathbf{A}_{ii} = 0$.
2. Form a diagonal matrix $\mathbf{D}$, where $\mathbf{D}_{ii} = \sum_j \mathbf{A}_{ij}$, and construct the matrix $\mathbf{L} = \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$.[3]
3. Find the k largest eigenvectors of $\mathbf{L}$, u_1, u_2, ..., u_k, and form the matrix $\mathbf{U} = [u_1 u_2 ... u_k] \in \mathbb{R}^{n \times k}$.
4. Normalize $\mathbf{U}$, such that $\mathbf{U}_{ij} = \mathbf{U}_{ij} / (\sum_j \mathbf{U}_{ij}^2)^{1/2}$.
5. Apply the k-means algorithm on $\mathbf{U}$ to cluster the n points into k clusters.

Based on the above description, the standard spectral clustering algorithm clusters n points to k discrete indicators, which can be referred to as "discrete clustering". Ding and He [15] proved that the k principal components of a term-document co-occurrence matrix,

[3]In spectral graph theory [9] and Laplacian Eigenmaps [3], the Laplacian matrix $\widetilde{\mathbf{L}} = \mathbf{I} - \mathbf{L}$, where $\mathbf{I}$ is an identity matrix. The changes in these forms of Laplacian matrix will only change the eigenvalues (from λ_i to $1 - \lambda_i$) but have no impact on eigenvectors. Thus selecting the k smallest eigenvectors of $\widetilde{\mathbf{L}}$ in [9, 3] is equivalent to selecting the k largest eigenvectors of $\mathbf{L}$ in this paper.

which are referred to as the k largest eigenvectors u_1, u_2, ..., u_k in step 3, are actually the continuous solution of the cluster membership indicators of documents in the k-means clustering method. More specifically, the k principal components can automatically perform data clustering in the subspace spanned by the k principle components. This implies that a mapping function constructed from the k principal components can cluster original data and map them to a new space spanned by the clusters simultaneously. Motivated by this discovery, we show how to adapt the spectral clustering algorithm for cross-domain feature alignment.

Given the feature bipartite graph G, our goal is to learn a feature alignment mapping function $\varphi(\cdot) : \mathbb{R}^{m-l} \to \mathbb{R}^k$, where m is the number of all features, l is the number of domain-independent features and $m - l$ is the number of domain-specific features.

1. Form a weight matrix $\mathbf{M} \in \mathbb{R}^{(m-l)\times l}$, where $\mathbf{M}_{ij}$ corresponds to the co-occurrence relationship between a domain-specific word $w_i \in W_{DS}$ and a domain-independent word $w_j \in W_{DI}$.
2. Form an affinity matrix $\mathbf{A} = \begin{bmatrix} \mathbf{0} & \mathbf{M} \\ \mathbf{M}^T & \mathbf{0} \end{bmatrix} \in \mathbb{R}^{m\times m}$ of the bipartite graph, where the first $m - l$ rows and columns correspond to the $m-l$ domain-specific features, and the last l rows and columns correspond to the l domain-independent features.
3. Form a diagonal matrix $\mathbf{D}$, where $\mathbf{D}_{ii} = \sum_j \mathbf{A}_{ij}$, and construct the matrix $\mathbf{L} = \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$.
4. Find the k largest eigenvectors of $\mathbf{L}$, u_1, u_2, ..., u_k, and form the matrix $\mathbf{U} = [u_1 u_2 ... u_k] \in \mathbb{R}^{m\times k}$.
5. Define the feature alignment mapping function as $\varphi(x) = x\mathbf{U}_{[1:m-l,:]}$, where $\mathbf{U}_{[1:m-l,:]}$ denotes the first $m-l$ rows of $\mathbf{U}$ and $x \in \mathbb{R}^{1\times(m-l)}$.

Given a feature alignment mapping function $\varphi(\cdot)$, for a data example x_i in either a source domain or target domain, we can first apply $\phi_{DS}(\cdot)$ to identify the view associated with domain-specific features of x_i, and then apply $\varphi(\cdot)$ to find a new representation $\varphi(\phi_{DS}(x_i))$ of the view of domain-specific features of x_i. Note that the affinity matrix $\mathbf{A}$ constructed in Step 2 is similar to the affinity matrix of a term-document bipartite graph proposed in [14], which is used for spectral co-clustering terms and documents simultaneously. Though our goal is only to cluster domain-specific features, it is proved that clustering two related sets of points simultaneously can often get better results than only clustering one single set of points [14].

4.4 Feature Augmentation

If we have selected domain-independent features and aligned domain-specific features perfectly, then we can simply augment domain-independent features with the features learned by the feature alignment algorithm to generate a perfect representation for cross-domain sentiment classification. However, in practice, we may not be able to identify domain-independent features correctly and thus fail to perform feature alignment perfectly. Similar to the strategy used in [1, 6], we augment all original features with features learned by feature alignment to construct a new representation. A tradeoff parameter γ is used in this feature augmentation to balance the effect of original features and new features. So, for each data example x_i, the new feature representation is defined as

$$\widetilde{x}_i = [x_i,\ \gamma\varphi(\phi_{DS}(x_i))],$$

where $x_i \in \mathbb{R}^{1\times m}$, $\widetilde{x}_i \in \mathbb{R}^{1\times m+k}$ and $0 \le \gamma \le 1$. In practice, the value of λ can be determined by evaluation on some heldout data. The whole process of our proposed framework for cross-domain sentiment classification is presented in Algorithm 1.

Algorithm 1 Spectral Domain-Specific Feature Alignment for Cross-Domain Sentiment Classification

Input: labeled source domain data $\mathcal{D}_{src} = \{(x_{src_i}, y_{src_i})\}_{i=1}^{n_{src}}$, unlabeled target domain data $\mathcal{D}_{tar} = \{x_{tar_j}\}_{j=1}^{n_{tar}}$, the number of clusters K and the number of domain-independent features m.

Output: adaptive classifier $f : X \to Y$.

1: Apply the criteria mentioned in Section 4.1 on $\mathcal{D}_{src}$ and $\mathcal{D}_{tar}$ to select l domain-independent features. The remaining $m - l$ features are treated as domain-specific features.
$\Phi_{DI} = \begin{bmatrix} \phi_{DI}(x_{src}) \\ \phi_{DI}(x_{tar}) \end{bmatrix}$ and $\Phi_{DS} = \begin{bmatrix} \phi_{DS}(x_{src}) \\ \phi_{DS}(x_{tar}) \end{bmatrix}$.
2: By using Φ_{DI} and Φ_{DS}, calculate (*DI-word*)-(*DS-word*) co-occurrence matrix $\mathbf{M} \in \mathbb{R}^{(m-l)\times l}$.
3: Construct matrix $\mathbf{L} = \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$,
where $\mathbf{A} = \begin{bmatrix} \mathbf{0} & \mathbf{M} \\ \mathbf{M}^T & \mathbf{0} \end{bmatrix}$.
4: Find the K largest eigenvectors of $\mathbf{L}$, u_1, u_2, ..., u_K, and form the matrix $\mathbf{U} = [u_1 u_2 ... u_K] \in \mathbb{R}^{m\times K}$.
Let mapping $\varphi(x_i) = x_i\mathbf{U}_{[1:m-l,:]}$, where $x_i \in \mathbb{R}^{m-l}$
5: Return a classifier f, trained on
$\{([x_{src_i}\ \gamma\varphi(\phi_{DS}(x_{src_i}))], y_{src_i})\}_{i=1}^{n_{src}}$

5. EXPERIMENTS

In this section, we will describe our experiments on two real-world datasets and show the effectiveness of our proposed **SFA** for cross-domain sentiment classification.

5.1 Datasets

In this sub-section, we first describe the datasets used in our experiments. The first dataset is from Blitzer *et al.* [6]. It contains a collection of product reviews from Amazon. The reviews are about four product domains: **books** (**B**), **dvds** (**D**), **electronics** (**E**) and **kitchen appliances** (**K**). Each review is assigned a sentiment label, -1 (negative review) or $+1$ (positive review), based on the rating score given by the review author. In each domain, there are $1,000$ positive reviews and $1,000$ negative ones. In this dataset, we can construct 12 cross-domain sentiment classification tasks: **D** $\to$ **B**, **E** $\to$ **B**, **K** $\to$ **B**, **K** $\to$ **E**, **D** $\to$ **E**, **B** $\to$ **E**, **B** $\to$ **D**, **K** $\to$ **D**, **E** $\to$ **D**, **B** $\to$ **K**, **D** $\to$ **K**, **E** $\to$ **K**, where the word before an arrow corresponds with the source domain and the word after an arrow corresponds with the target domain. We use $RevDat$ to denote this dataset. The sentiment classification task on this dataset is document-level sentiment classification.

The other dataset is collected by us for experiment purpose. We have crawled a set of reviews from Amazon[4], Yelp[5] and Citysearch[6] websites. The reviews from Amazon are about three product domains: **video game** (**V**), **electronics** (**E**) and **software** (**S**). The reviews from Yelp and Citysearch are about the **hotel** (**H**) domain. Instead of assigning each review with a label, we split these reviews into sentences and manually assign a polarity label for each

[4]http://www.amazon.com/
[5]http://www.yelp.com/
[6]http://www.citysearch.com/

sentence. In each domain, we randomly select $1,500$ positive sentences and $1,500$ negative ones for experiment. Similarly, we also construct 12 cross-domain sentiment classification tasks: **V** → **H**, **V** → **E**, **V** → **S**, **S** → **E**, **S** → **V**, **S** → **H**, **E** → **V**, **E** → **H**, **E** → **S**, **H** → **S**, **H** → **E**, **S** → **V**. We use $SentDat$ to denote this dataset. Sentiment classification task on this dataset is sentence-level sentiment classification. For both datasets, we use Unigram and Bigram features to represent each data example (a review in $RevDat$ and a sentence in $SentDat$). The summary of the datasets is described in Table 5.

Table 5: Summary of Datasets Used for Evaluation.

Dataset	Domain	# Reviews	# Pos	# Neg	# Features
$RevDat$	dvds	$2,000$	$1,000$	$1,000$	$473,856$
	kitchen	$2,000$	$1,000$	$1,000$	
	electronics	$2,000$	$1,000$	$1,000$	
	books	$2,000$	$1,000$	$1,000$	
$SentDat$	video game	$3,000$	$1,500$	$1,500$	$287,504$
	hotel	$3,000$	$1,500$	$1,500$	
	software	$3,000$	$1,500$	$1,500$	
	electronics	$3,000$	$1,500$	$1,500$	

5.2 Baselines

In order to investigate the effectiveness of our method, we have compared it with several algorithms. In this sub-section, we describe some baseline algorithms with which we compare **SFA**. One baseline method denoted by **NoTransf**, is a classifier trained directly with the source domain training data. The gold standard (denoted by **upperBound**) is an in-domain classifier trained with labeled data from the target domain. For example, for **D** → **B** task, **NoTransf** means that we train a classifier with labeled data of **D** domain. **upperBound** corresponds with a classifier trained with the labeled data from **B** domain. So, the performance of **upperBound** in **D** → **B** task can be also regarded as an upper bound of **E** → **B** and **K** → **B** tasks. Another baseline method denoted by **LSA** is a classifier trained on a new representation which augments original features with new features which are learned by applying latent semantic analysis (also can be referred to as principal component analysis) [13] on the original view of domain-specific features (as shown in Table 2). A third baseline method denoted by **FALSA** is a classifier trained on a new representation which augments original features with new features which are learned by applying latent semantic analysis on the co-occurrence matrix of domain-independent and domain-specific features. We compare our method with **LSA** and **FALSA** in order to investigate if spectral feature clustering is effective in aligning domain-specific features. We have also compared our algorithm with a method: structural correspondence learning (**SCL**) proposed in [6]. We follow the details described in Blitzer's thesis [5] to implement **SCL** with logistic regression to construct auxiliary tasks. Note that **SCL**, **LSA**, **FALSA** and our proposed **SFA** all use unlabeled data from the source and target domains to learn a new representation and train classifiers using the labeled source domain data with new representations.

5.3 Parameter Settings & Evaluation Criteria

For **NoTransf**, **upperBound**, **LSA**, **FALSA**and **SFA**, we use logistic regression as the basic sentiment classifier. The library implemented in [16] is used in all our experiments. The tradeoff parameter C in logistic regression [16] is set to be $10,000$, which is equivalent to set $\lambda = 0.0001$ in [5]. The parameters of each model are tuned on some heldout data in **E** → **B** task of $RevDat$ and **H** → **S** task of $SentDat$, and are fixed to be used in all experiments. We use accuracy to evaluate the sentiment classification result: the percentage of correctly classified examples over all testing examples. The definition of accuracy is given as follows,

$$Accuracy = \frac{|\{x|x \in \mathcal{D}_{tst} \cap f(x) = y\}|}{|\{x|x \in \mathcal{D}_{tst}\}|},$$

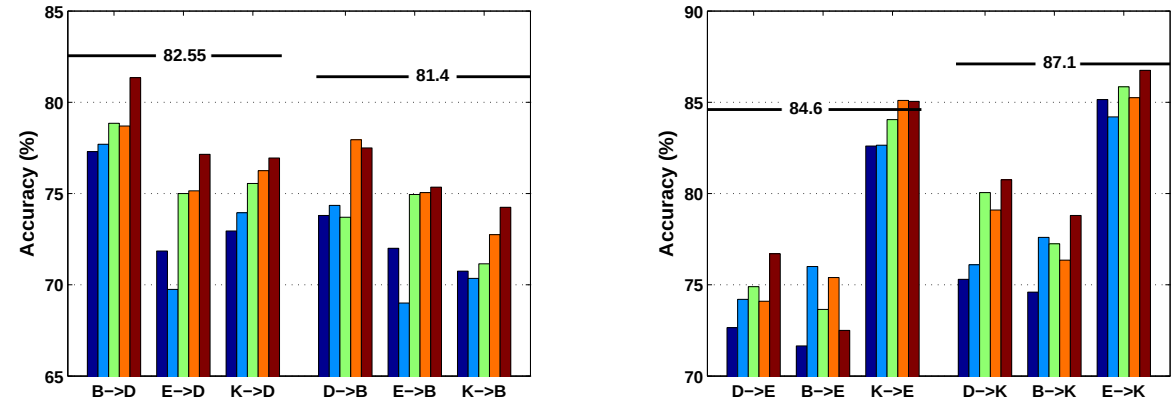
where $\mathcal{D}_{tst}$ denotes the test data, y is the ground truth sentiment polarity and $f(x)$ is the predicted sentiment polarity. For all experiments on $RevDat$, we randomly split each domain data into a training set of 1,600 instances and a test set of 400 instances. For all experiments on $SentDat$, we randomly split each domain data into a training set of 2,000 instances and a test set of 1,000 instances. The evaluation of cross-domain sentiment classification methods is conducted on the test set in the target domain without labeled training data in the same domain. We report the average results of 5 random times.

5.4 Overall Comparison Results

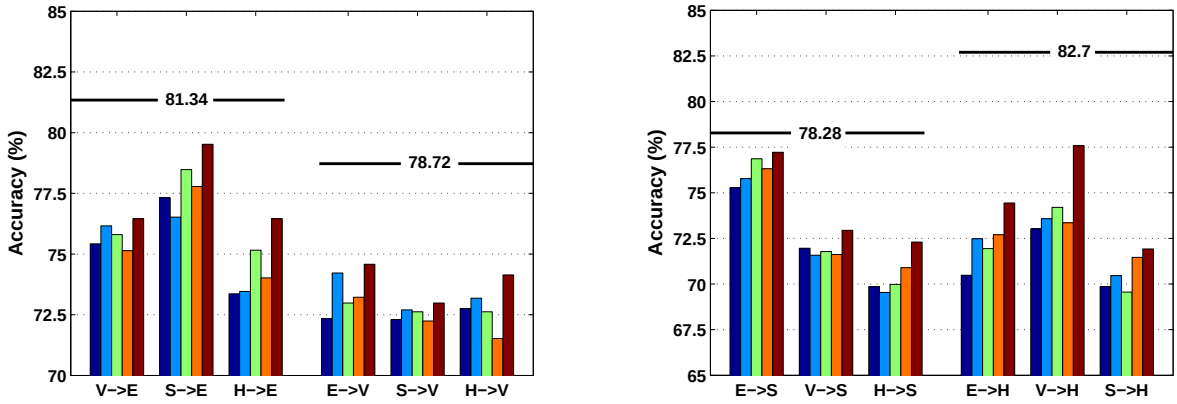
In this section we compare the accuracy of **SFA** with **NoTransf**, **LSA**, **FALSA** and **SCL** by 24 tasks on two datasets. For **LSA**, **FALSA**and **SFA**, we use Eqn.(1) defined in Section 4.1 to identity domain-independent and domain-specific features. We adopt the following settings: the number of domain-independent features $l = 500$, the number of domain-specific features clusters $k = 100$ and the parameter in feature augmentation $\gamma = 0.6$. Studies of the **SFA** parameters are presented in Section 5.5 and 5.6. For **SCL**, we use mutual information to select "pivots". The number of "pivots" is set to be 500, and the number of dimensionality h in [6] is set to be 50. All these parameters and domain-independent feature (or "pivot") selection methods are determined based on results on the heldout data mentioned in the previous section.

Figure 2(a) shows the comparison results of different methods on $RevDat$. In the figure, each group of bars represents a cross-domain sentiment classification task. Each bar in specific color represents a specific method. The horizontal lines are accuracies of **upperBound**. From the figure, we can observe that the four domains of $RevDat$ can be roughly classified into two groups: **B** and **D** domains are similar to each other, as are **K** and **E**, but the two groups are different from each other. Adapting a classifier from **K** domain to **E** domain is much easier than adapting it from **B** domain. Clearly, our proposed **SFA** performs better than other methods including state-of-the-art method **SCL** in most tasks. As mentioned in Section 3, clustering domain-specific features with bag-of-words representation may fail to find a meaningful new representation for cross-domain sentiment classification. Thus **LSA** only outperforms **NoTransf** slightly in some tasks, but its performance may even drop on other tasks. It is not surprising to find that **FALSA** gets significant improvement compared to **NoTransf** and **LSA**. The reason is that representing domain-specific features via domain-independent features can reduce the gap between domains and thus find a reasonable representation for cross-domain sentiment classification. Our proposed **SFA** can not only utilize the co-occurrence relationship between domain-independent and domain-specific features to reduce the gap between domains, but also use graph spectral clustering techniques to co-align both kinds of features to discover meaningful clusters for domain-specific features. Though our goal is only to cluster domain-specific features, it has been proved that clustering two related sets of points simultaneously can often get better results than clustering one single set of points only [14].

From the comparison results on $SentDat$ shown in Figure 2(b), we can get similar conclusion: **SFA** outperforms other methods



(a) Comparison Results on $RevDat$.



(b) Comparison Results on $SentDat$.

Figure 2: Comparison Results (unit: %) on Two Datasets.

in most tasks. One interesting observation from the results is that **SCL** does not work well compared to its performance on $RevDat$. One reason may be that in sentence-level sentiment classification, the data is quite sparse. In this case, it is hard to construct a reasonable number of auxiliary tasks that are useful to model the relationship between "pivots" and "non-pivots". The performance of **SCL** highly relies on the auxiliary tasks. Thus in this dataset, **SCL** even performs worse than **FALSA** in some tasks. We do **t-test** on the comparison results of the two datasets and find that **SFA** outperforms other methods with 0.95 confidence intervals.

5.5 Effect of Domain Independent Features

In this section, we conduct two experiments to study the effect of domain-independent features on the performance of **SFA**. The first experiment is to test the effect of domain-independent features identified by different methods on the overall performance of **SFA**. The second one is to test the effect of different numbers of domain-independent features on **SFA** performance. As mentioned in Section 4.1, besides using Eqn. (1) to identify domain-independent and domain-specific features, we can also use the other two strategies to identify them. In Table 6, we summarize the comparison results of **SFA** using different methods to identify domain-independent features. We use $\textbf{SFA}_{DI}$, $\textbf{SFA}_{FQ}$ and $\textbf{SFA}_{MI}$ to denote **SFA** using Eqn. (1), frequency of features in both domains and mutual information between features and labels in the source domain respectively. From the table, we can observe that $\textbf{SFA}_{DI}$ and $\textbf{SFA}_{FQ}$ achieve comparable results and they are stable in most tasks. While $\textbf{SFA}_{MI}$ may work very well in some tasks such as **K** $\rightarrow$ **D** and **E** $\rightarrow$ **B** of $RevDat$, but work very bad in some tasks such as **E** $\rightarrow$ **D** and **D** $\rightarrow$ **E** of $RevDat$. The reason is that applying mutual information on source domain data can find features that are relevant to the source domain labels but cannot guarantee the selected features to be domain independent. In addition, the selected features may be irrelevant to the labels of the target domain. To test the effect of the number of domain-independent features on the performance of **SFA**, we apply **SFA** on 12 tasks randomly selected from the two datasets, and fix $k = 100$, $\gamma = 0.6$. The value of l is changed from 300 to 700 with step length 100. The results are shown in Figure 3(a) and 3(b). From the figures, we can find that when l is in the range of $[400, 700]$, **SFA** performs well and stably in most tasks. Thus **SFA** is robust with regard to the quality and numbers of domain-independent features.

Table 6: Experiments with Different Domain-Independent Feature Selection Methods. Numbers in the table are accuracies in percentage.

$RevDat$												
	B→D	**E→D**	**K→D**	**D→B**	**E→B**	**K→B**	**D→E**	**B→E**	**K→E**	**D→K**	**B→K**	**E→K**
SFA_{DI}	81.35	77.15	76.95	77.5	75.65	74.8	76.7	72.5	85.05	80.75	78.8	86.75
SFA_{FQ}	81.25	77	76.6	78.25	75.35	74.25	76.05	73.45	84.9	80.6	79.05	85.8
SFA_{MI}	80.1	70.4	78.45	79.8	78.25	75.15	70.85	73	82.05	78.9	78.8	86.75
$SentDat$												
	V→E	**S→E**	**H→E**	**E→V**	**S→V**	**H→V**	**E→S**	**V→S**	**H→S**	**E→H**	**V→H**	**S→H**
SFA_{DI}	76.64	79.52	76.46	74.58	72.98	74.14	77.22	72.94	72.3	74.44	77.58	71.92
SFA_{FQ}	76.62	79.5	76.64	74.38	73.16	74.54	77.5	72.96	72.22	75	77.46	71.62
SFA_{MI}	76.96	79.08	76.46	75.06	73.86	74.88	77.48	73.22	72.38	75.98	77.08	72.46

5.6 Parameter Sensitivity

Besides the number of domain-independent features l, there are two other parameters in **SFA**: one of them is the number of domain-specific feature-clusters k and the other is the tradeoff parameter λ in feature augmentation. In this section, we further test the sensitivity of these two parameters on the overall performance of **SFA**.

We first test the sensitivity of the parameter k. In this experiment, we fix $l = 500$, $\gamma = 0.6$ and change the value of k from 50 to 200 with step length 25. Figure 3(c) and 3(d) show the results of **SFA** under varying values of k. Note that when the cluster number k falls in the range from 75 to 175, **SFA** performs well and stably.

Finally, we test the sensitivity of the parameter γ. In this experiment, we fix $l = 500$, $k = 100$ and change the values of γ from 0.1 to 1 with step length 0.1. Results are shown in Figure 3(e) and 3(f). Apparently, when $\gamma \geq 0.3$, **SFA** works stably in most tasks.

6. RELATED WORK

Sentiment classification aims to predict the sentiment polarity of text data, e.g., text sentences and review articles, etc. It has drawn much research attention recently. Many machine learning techniques have been proposed for sentiment classification, such as unsupervised learning techniques [32], supervised learning techniques [30], graph-based semi-supervised learning techniques [17, 31], and matrix factorization techniques with lexical prior knowledge [23]. However, most sentiment classifiers are domain dependent. It is challenging to adapt a classifier trained on one domain to another domain. To address this problem, Blitzer *et al.* [6] proposed the **SCL** algorithm to exploit domain adaptation techniques for sentiment classification. **SCL** is motivated by a multi-task learning algorithm, alternating structural optimization (**ASO**), proposed by Ando and Zhang [1]. **SCL** tries to construct a set of related tasks to model the relationship between "pivot features" and "non-pivot features". Then "non-pivot features" with similar weights among tasks tend to be close with each other in a low-dimensional latent space. However, in practice, it is hard to construct a reasonable number of related tasks from data (as shown in Section 5.4) which may limit the transfer ability of **SCL** for cross-domain sentiment classification. More recently, Li *et al.* [22] proposed to transfer common lexical knowledge across domains via matrix factorization techniques.

Domain adaptation can be referred to as a special setting of transfer learning [28], which aims at transferring knowledge across domains or tasks. Besides sentiment classification, domain adaptation techniques have been widely applied to other Web applications, such as text classification [11, 8, 33, 10], part of speech tagging [2, 7, 20, 12], named-entity recognition and shallow parsing [12]. Most existing domain adaptation methods can be classified into two categories: feature-representation adaptation [11, 8, 33, 2, 7, 12] and instance-weight adaptation [20]. The basic idea of the first kind of methods is to develop an adaptive feature representation that is effective in reducing the difference between domains. Among these works, the method proposed by Dai *et al.* [10] is also based on graph spectral techniques. But the bipartite graph constructed in [10] is among features, instances and tasks. While in this work, we build a bipartite graph between domain-independent and domain-dependent features. Instead of constructing new feature representations, instance-weight approaches assume that some training data in the source domain are very useful for the target domain and these data can be used to train model for the target domain after re-weighting. Theoretical analysis of domain adaptation has also been studied in [4].

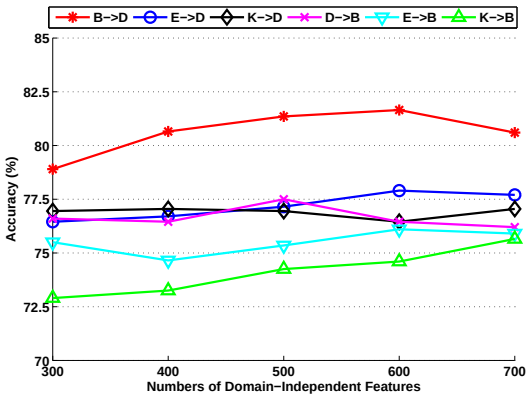
7. CONCLUSION

In this paper, we propose a general framework for cross-domain sentiment classification. In our framework, we first build a bipartite graph between domain-independent and domain-specific features. Then, we propose a spectral feature alignment (**SFA**) algorithm to align the domain-specific words from the source and target domains into meaningful clusters, with the help of domain-independent words as a bridge. In this way, the clusters can be used to reduce the gap between domain specific words of the two domains, which is helpful for training an accurate classifier for the target domain. Our experimental results on both document-level and sentence-level sentiment classification tasks demonstrate the effectiveness of our proposed framework.

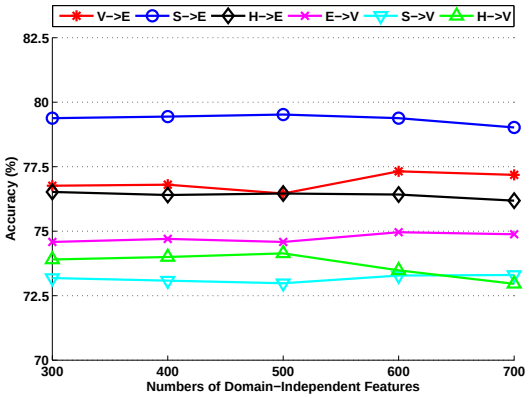
In the future, we are planning to encode some lexical knowledge from the source domain to the spectral domain-specific feature alignment framework. The reason is that each word has its polarity category. If we get the polarity knowledge of some words, we can adopt semi-supervised learning techniques to help learn more reasonable clusters of domain-specific features from the bipartite graph. In addition, we are planning to develop a more effective feature selection method to identify domain-independent features. Finally, we are also planning to extend our proposed **SFA** to solve sentiment classification problems from multiple source domains.

8. ACKNOWLEDGMENTS

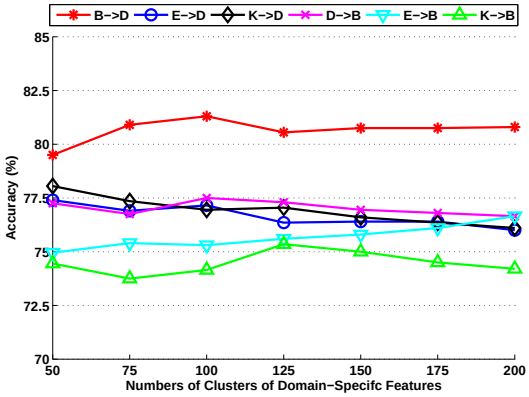
Sinno J. Pan and Qiang Yang thank a grant from Microsoft Research Asia MRA08/09.EG03 and Hong Kong CERG/China-NSFC Grant N_HKUST624/09 for their support. We aslo thank John Blitzer and Yangsheng Ji for comments on implementation of the SCL algorithm and Evan W. Xiang for providing toolkits to preprocess the datasets.



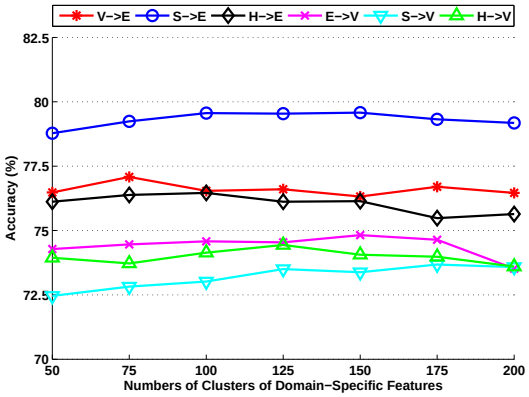
(a) Results on $RevDat$ under Varying Numbers of Domain-Independent Features.



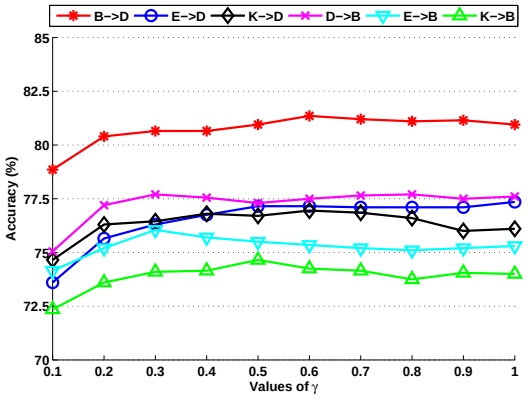
(b) Results on $SentDat$ under Varying Numbers of Domain-Independent Features.



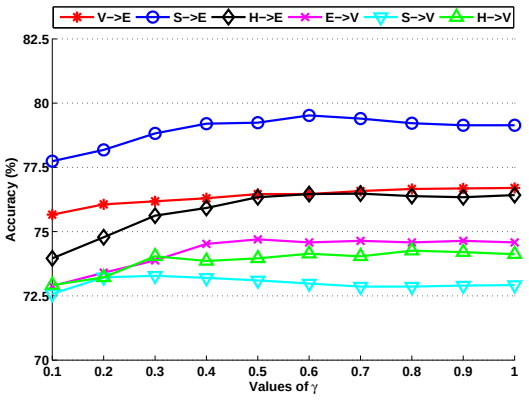
(c) Results on $RevDat$ under Varying Numbers of Feature-Clusters.



(d) Results on $SentDat$ under Varying Numbers of Feature-Clusters.



(e) Results on $RevDat$ under Varying Values of γ.



(f) Results on $SentDat$ under Varying Values of γ.

Figure 3: Parameter Sensitivity Study of SFA on Two Datasets.

9. REFERENCES

[1] R. K. Ando and T. Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6:1817–1853, 2005.

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

[3] M. Belkin and P. Niyogi. Laplacian eigenmaps for

dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396, 2003.

[4] S. Ben-David, J. Blitzer, K. Crammer, and F. Pereira. Analysis of representations for domain adaptation. In *Annual Conference on Neural Information Processing Systems 19*, pages 137–144, Cambridge, MA, 2007. MIT Press.

[5] J. Blitzer. *Domain Adaptation of Natural Language Processing Systems*. PhD thesis, The University of Pennsylvania, 2007.

[6] J. Blitzer, M. Dredze, and F. Pereira. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 432–439, Prague, Czech Republic, 2007.

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

[8] B. Chen, W. Lam, I. W. Tsang, and T.-L. Wong. Extracting discriminative concepts for domain adaptation in text mining. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 179–188, New York, NY, USA, 2009. ACM.

[9] F. R. K. Chung. *Spectral Graph Theory*. Number 92 in CBMS Regional Conference Series in Mathematics. American Mathematical Society, 1997.

[10] W. Dai, O. Jin, G.-R. Xue, Q. Yang, and Y. Yu. Eigentransfer: a unified framework for transfer learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 25–31, Montreal, Quebec, Canada, June 2009.

[11] W. Dai, G. Xue, Q. Yang, and Y. Yu. Co-clustering based classification for out-of-domain documents. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Jose, California, USA, August 2007.

[12] H. Daumé III. Frustratingly easy domain adaptation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 256–263, Prague, Czech Republic, June 2007.

[13] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41:391–407, 1990.

[14] I. S. Dhillon. Co-clustering documents and words using bipartite spectral graph partitioning. In *Proceedings of the 7th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 269–274. ACM, 2001.

[15] C. Ding and X. He. K-means clustering via principal component analysis. In *Proceedings of the twenty-first international conference on Machine learning*, pages 225–232, Banff, Alberta, Canada, 2004. ACM.

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

[17] A. Goldberg and X. Zhu. Seeing stars when there aren't many stars: Graph-based semi-supervised learning for sentiment categorization. In *Proceedings of TextGraphs: the 1st Workshop on Graph Based Methods for Natural Language Processing*, pages 45–52. ACL, June 2006.

[18] J. A. Hartigan and M. A. Wong. A k-means clustering algorithm. *Applied Statistics*, 28:100–108, 1979.

[19] M. Hu and B. Liu. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177, Seattle, WA, USA, 2004. ACM.

[20] J. Jiang and C. Zhai. Instance weighting for domain adaptation in nlp. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 264–271, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

[21] N. Jindal and B. Liu. Opinion spam and analysis. In *Proceedings of the international conference on Web search and web data mining*, pages 219–230, Palo Alto, California, USA, 2008. ACM.

[22] T. Li, V. Sindhwani, C. Ding, and Y. Zhang. Knowledge transformation for cross-domain sentiment classification. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 716–717, Boston, MA, USA, 2009. ACM.

[23] T. Li, Y. Zhang, and V. Sindhwani. A non-negative matrix tri-factorization approach to sentiment classification with lexical prior knowledge. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the Association of Computational Linguistics*, pages 244–252, Suntec, Singapore, August 2009. ACL.

[24] B. Liu. *Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data*. Springer, January 2007.

[25] Y. Lu and C. Zhai. Opinion integration through semi-supervised topic modeling. In *Proceedings of the 17th International Conference on World Wide Web*, pages 121–130, Beijing, China, April 2008. ACM.

[26] Y. Lu, C. Zhai, and N. Sundaresan. Rated aspect summarization of short comments. In *Proceedings of the 18th international conference on World wide web*, pages 131–140, Madrid, Spain, 2009. ACM.

[27] A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems 14*, pages 849–856, 2001.

[28] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 2009. Available at `http://doi.ieeecomputersociety.org/10.1109/TKDE.2009.191`.

[29] B. Pang and L. Lee. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1-2):1–135, 2008.

[30] B. Pang, L. Lee, and S. Vaithyanathan. Thumbs up? Sentiment classification using machine learning techniques. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 79–86, 2002.

[31] V. Sindhwani and P. Melville. Document-word co-regularization for semi-supervised sentiment analysis. In *Proceedings of the 8th IEEE International Conference on Data Mining*, pages 1025–1030, Washington, DC, USA, 2008. IEEE Computer Society.

[32] P. Turney. Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the 40th Annual Meeting fo the Association for Computational Linguistics*, pages 417–424. ACL, 2002.

[33] S. Xie, W. Fan, J. Peng, O. Verscheure, and J. Ren. Latent space domain transfer between high dimensional overlapping distributions. In *18th International World Wide Web Conference*, pages 91–100, April 2009.