Transfer Learning via Feature Isomorphism Discovery

Shimin Di
Hong Kong University of Science and Technology
Hong Kong SAR, China
sdiaa@cse.ust.hk

Jingshu Peng
Hong Kong University of Science and Technology
Hong Kong SAR, China
jpengab@cse.ust.hk

Yanyan Shen
Shanghai Jiao Tong University
Shanghai, China
shenyy@sjtu.edu.cn

Lei Chen
Hong Kong University of Science and Technology
Hong Kong SAR, China
leichen@cse.ust.hk

ABSTRACT

Transfer learning has gained increasing attention due to the inferior performance of machine learning algorithms with insufficient training data. Most of the previous homogeneous or heterogeneous transfer learning works aim to learn a mapping function between feature spaces based on the inherent correspondence across the source and target domains or labeled instances. However, in many real world applications, existing methods may not be robust when the correspondence across domains is noisy or labeled instances are not representative. In this paper, we develop a novel transfer learning framework called *Transfer Learning via Feature Isomorphism Discovery* (abbreviated to TLFid), which owns high tolerance for noisy correspondence between domains as well as scarce or non-existing labeled instances. More specifically, we propose a feature isomorphism approach to discovering common substructures across feature spaces and learning a feature mapping function from the target domain to the source domain. We evaluate the performance of TLFid on the cross-lingual sentiment classification tasks. The results show that our method achieves significant improvement in terms of accuracy compared with the state-of-the-art methods.

CCS CONCEPTS

• **Computing methodologies** → **Transfer learning**; • **Applied computing** → **Document searching**; • **Information systems** → *Data mining*;

KEYWORDS

Transfer learning; cross-lingual; subgraph isomorphism

ACM Reference Format:
Shimin Di, Jingshu Peng, Yanyan Shen, and Lei Chen. 2018. Transfer Learning via Feature Isomorphism Discovery. In *KDD '18: The 24th ACM SIGKDD International Conference on Knowledge Discovery Data Mining, August 19–23, 2018, London, United Kingdom.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3219819.3220029

1 INTRODUCTION

Recently, due to the huge success of Alphago and other applications such as Texas Poker games, machine learning has attracted great attention in both academia and industry. However, the effectiveness of machine learning algorithms typically depend on the availability of labeled data. In the applications such as image classification and many Natural Language Processing (NLP) tasks, people have to do a large amount of repetitive work on labeling training data in a particular domain. Sometimes it would be extremely hard to apply machine learning techniques if it is impossible to collect enough labeled training data [19]. In order to address this problem, transfer learning has been widely adopted to transfer knowledge from a source domain with labeled instances to a target domain with scarce or even no labels.

Regardless of homogeneous or heterogeneous transfer learning, most of previous efforts attempt to learn a feature mapping function which is able to project features from the source domain onto the target domain. In such a way, labeled instances can be easily mapped from the source domain to the target domain. In other words, the source domain which has sufficient labeled data could be used to train a classifier for the target domain. However, most existing works of feature mapping either rely on the correspondence among domains [6, 24] or labeled instances in both domains [8, 14]. When there is noisy or even no correspondence between feature spaces, prior methods based on the cross-domain correspondence may encounter negative transfer effects. As for those works that learn feature mapping functions from labeled instances, they assume that instances are representative in both the source and target domains. However, this assumption may not hold in real scenarios. The performance of such algorithms can hardly be guaranteed due to the fact that the labeled data might be biased. To be more specific, the available labeled instances may fail to derive a feature mapping function that reflects the true distribution of the feature space. This problem will be further discussed in Section 3.

In the recent years, word embedding [17] has helped improve the performance of many NLP tasks such as entity recognition and sentiment analysis. Word embedding represents words from the corpus to vectors of real numbers. In other words, it transforms the representation of each word from one dimension space to a continuous vector space. Consider the most famous example: "$queen \approx king - man + woman$"[17], where word embedding is capable of capturing both syntactic and semantic meanings of words in the vector space by exploring the correlation among words. Inspired by the embedding methods, we tend to investigate whether we can apply the embedding technique to represent the correlation among features in one feature space. Note that our solution is not limited to transfer learning over text-oriented tasks, but also works for the image tasks by extracting image features and representing each image by a sequential feature vector. As mentioned before, in transfer learning tasks, feature spaces in different domains may have low correspondence. But various feature spaces probably own common feature structures, which means that some features may have similar behaviors in their respective feature spaces, though the distributions of different feature spaces may not be close to each other.

Motivated by the above observations, we propose a novel transfer learning framework named Transfer Learning via Feature Isomorphism Discovery (TLFid) which is robust, no matter whether there is low correspondence across domains or there only exist scarce labeled instances in the target domain. Specifically, the TLFid framework consists of three major components: 1) extracting features from source and target domains respectively, 2) learning a feature mapping function across domains through discovering the common feature structures, and 3) reconstructing feature representations to build a target classifier effectively. The main contributions of our work can be summarized as follows:

- We propose a novel transfer learning framework named TLFid to explore the common feature structures in both source and target domains based on the embedding techniques.
- TLFid is effective to learn the feature mapping function with the help of feature subgraph isomorphism discovery, when the feature spaces in the source and target domains have noisy or low correspondence, or the target domain has few labeled instances.
- We conduct experiments to verify the effectiveness of TLFid. The results show that TLFid outperforms the state-of-the-art methods on the cross-lingual text sentiment classification task.

The remainder of the paper is organized as follows. We review the related works in Section 2. We present basic notions and the problem definition in Section 3. After that, we describe the technical details of TLFid in Section 4. In Section 5, we validate the effectiveness of TLFid with extensive experiments. We conclude this paper and discuss some future works in Section 6.

2 RELATED WORK

Homogeneous transfer learning assumes that features in different domains follow the same distribution. It is essential to make use of high correspondence between two feature spaces. The self-taught learning [22] tries to learn the latent feature factors from a source domain and applies to a target domain directly. TCA [18] learns the latent feature factors shared across domains with the objective of minimizing the difference between domains.

Different from homogeneous transfer learning, heterogeneous transfer learning aims to transfer knowledge between domains with different feature spaces. HeMap [24] projected both source and target instances into one latent space to preserve the original structures of domains as much as possible and obtained small distances between projections of instance features. However, this approach requires the existence of strong correspondence in heterogeneous features across domains. TLRisk [6] built a feature translator to transfer knowledge between different feature spaces via minimizing risk of transfer. However, TLRisk may not be robust when the translator is affected by the noisy correspondence among heterogeneous features. In contrast to these correspondence based methods, MMDT [14] does not require the correspondence between domains. Instead, it requires the existence of sufficient labeled instances in the target domain. HFA [8] was proposed to project both domains into one latent space which is later augmented by the original features. Nevertheless, HFA can only make use of the labeled examples without considering the large number of unlabeled instances.

One recent work [26] proposed a method based on manifold alignment, named heterogeneous domain Adaptation with Manifold Alignment (DAMA). The key idea is to align different domains into a latent space using label information or correspondence between two domains. However, DAMA assumed that the correspondence matrix between heterogeneous features across domains is binary and cannot handle heterogeneous label spaces. In this paper, we propose a feature isomorphism method to map features from the target domain to the source domain, which performs well when noisy correspondence exists across domains or scarce labels exist in the target domain.

3 PROBLEM FORMULATION

In this section, we introduce some notations used throughout this paper and formulate our problem.

We have labeled instances in the source domain $\mathcal{S} = \{(x_i^s, y_i^s)\}_{i=1}^{\ell_s}$ and unlabeled data in the target domain $\mathcal{T} = \{x_i^t\}_{i=1}^{\ell_t}$. Suppose that two collections of features $E_s = \{e_1^s, e_2^s, \cdots, e_{m_s}^s\}$ and $E_t = \{e_1^t, e_2^t, \cdots, e_{m_t}^t\}$ have been extracted from the source and target domains respectively with the help of embedding methods. Each input instance $x^s \in \{x_i^s\}_{i=1}^{\ell_s}$ or $x^t \in \{x_i^t\}_{i=1}^{\ell_t}$ can be encoded into a sequence vector in the feature space of E_s or E_t. We denote by $s(\cdot, \cdot)$ the evaluation metric of feature similarity in a specific domain.

Definition 3.1 (Feature correlation matrix). Let $C(E) \in R^{m \times m}$ be the correlation matrix based on the feature collection $E = \{e_1, \cdots, e_m\}$, where the $(i, j)^{th}$ element of $C(E)$ represents the feature similarity $s(e_i, e_j)$. Note that the feature correlation matrix is symmetric: $s(e_i, e_j) = s(e_j, e_i)$.

As discussed before, instead of mining the latent factor space shared by two domains, we are interest in discovering similar features which behave similarly in their own domains. After embedding features in E_s and E_t into the vector spaces, we first compute the similarities between pairwise features and then represent them with two feature correlation matrices $C(E_s)$ and $C(E_t)$.

Definition 3.2 (Feature mapping). Given two feature sets E_s and E_t where $|E_s| = |E_t| = m$, a feature mapping function π is a bijective mapping such that $\pi : \{e_1^t, \cdots, e_m^t\} \to \{e_1^s, \cdots, e_m^s\}$.

Now we are ready to discuss the idea of how a bijective feature mapping function can benefit from the feature correlation matrix. For ease of illustration, we first provide some examples. It is intuitive that we can map “attraktiv” in German to “attractive” by the shared latent space which indicates the semantic meaning of words such as $\pi(\text{“}attraktiv\text{”}) = \text{“}attractive\text{”}$. But such methods may not be effective when the correspondence among domains are biased [28]. For instance, the word “马上” in Chinese has two meanings: the first meaning is “immediately” and the second meaning is “horseback”. It is hard to know the exact meaning of this word due to various possible feature spaces. But it can be easily distinguished if we consider the relation of this word to other words in the same corpus or same feature space. For example, there is a high possibility that “马上” is related to “immediately” if “马上” has a positive correlation with the word “急忙” (hurriedly).

Following the above discussion, the feature correlation matrix has the potential to solve the problem when noisy correspondence exists in two domains. Also, the function π will not be biased by unrepresentative instances since we can obtain feature correlations from the entire two domains and surpass the limitation of scarce labeled instances. Moreover, the performance and role of one feature, i.e., one word may have subtle variance in different domains. For instance, as the word “马上” with the meaning "horseback" only appears in very limited Chinese classical writing corpus. In other words, the bijective function is sufficient to map features from a specific source domain to another domain. In general, we can utilize such information of feature correlations to transfer knowledge across domains. We next explain how to build the feature mapping by pivoting feature correlation matrices.

Definition 3.3 (Permutation matrix). Given a feature mapping function π between E_s and E_t, a $m \times m$ permutation matrix P_π is obtained by permuting the columns of the identity matrix I_m. That is, for each i, $p_{ij} = 1$ if $e_i^t = \pi(e_j^s)$ and 0 otherwise.

$$E_s = \begin{bmatrix} 1 & 0.8 & 0.9 \\ 0.8 & 1 & -0.6 \\ 0.9 & -0.6 & 1 \end{bmatrix} E_t = \begin{bmatrix} 1 & 0.9 & -0.6 \\ 0.9 & 1 & 0.8 \\ -0.6 & 0.8 & 1 \end{bmatrix}$$

For sake of understanding, we give a simple example here to illustrate $C(E)$, π and P_π. Suppose there are two feature correlation matrices E_s and E_t as shown above, it is easy to know that there is a permutation matrix P_π:

$$P_\pi = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix},$$

where $E_t = P_\pi E_s P_\pi^T$. Therefore, we can obtain a feature mapping function π in the two-line form: $\begin{pmatrix} e_1^s & e_2^s & e_3^s \\ e_2^t & e_3^t & e_1^t \end{pmatrix}$ such as $\pi(e_1^s) = e_2^t$.

In practice, it is possible that we cannot find an appropriate π that makes E_a equal to $P_\pi E_b P_\pi^T$ exactly due to various feature distributions. Also, we are not able to build a bijective mapping function across the source and target domains when $|E_s| \neq |E_t|$. As mentioned before, some features or instances may not be representative in a specific feature space. This leads us to only consider a part of features in one domain. It is natural to come up with the idea to only build the feature mapping function based on a common feature structure across domains. Given $\widetilde{E}_s \subset E_s$ and $\widetilde{E}_t \subset E_t$, TLFid aims to learn such a mapping function π by solving the optimization problem as follows:

$$\begin{aligned} &\arg\min_{\pi, \widetilde{E}_s, \widetilde{E}_t} \left\| P_\pi C(\widetilde{E}_t) P_\pi^T - C(\widetilde{E}_s) \right\|_F, \\ &\text{s.t.} |\widetilde{E}_s| = |\widetilde{E}_t| \geq threshold, \end{aligned} \tag{1}$$

where $\|\cdot\|_F$ denotes the Frobenius norm of matrix. Finally, with the help of π, we can map source features into the target feature space and train a model f_* on $\mathcal{S}$ which performs well on $\mathcal{T}$.

4 TRANSFER LEARNING VIA FEATURE ISOMORPHISM DISCOVERY

An intuitive way to solve Equation 1 in Section 3 is to list all the possible permutation matrices, but time complexity of the brute force algorithm is $O(n!)$. In this section, we propose to solve the optimization problem by finding the feature subgraph isomorphism across feature spaces. TLFid is based on the *Graph Isomorphism* that can search the common feature subgraphs between two feature correlation matrices and make use of the label information at the same time.

4.1 Extracting Feature Correlation Matrix

One of the most important tasks in transfer learning is to construct a good feature mapping function that is able to transfer most useful knowledge from the source domain to the target domain. Accordingly, before building feature mapping, the first step is to extract feature representations that can reflect the true correlation among features in their respective domains. Fortunately, various feature extracting methods have been explored. For instance, scale-invariant feature transform [15] was proposed to extract features from images while every word can be represented naturally as a feature in the text corpus.

Now we explain how to obtain feature correlation matrix with the embedding methods. Recently, word embedding [2] has showed great performance by representing words in corpus to vectors. Not only word embedding benefits various NLP tasks, but it also demonstrates high adaptability on the tasks associated with the sequence data, e.g., node2vec [12] applies similar idea to clustering tasks on graphs. Thanks to the success of exploring word semantics achieved by word embedding, it is natural to reflect the inherent correlation between features by considering each word as a unique feature. We begin with a set of instances without labels $\mathcal{X} = \{x_i\}_{i=1}^n$ and extract features $E = \{e_i\}_{i=1}^m$ from one domain. Instead of just using bag-of-word to encode instances, we represent each instance x_i to a variable-length sequence of tokens $x_i = (x_i^1, \cdots, x_i^T)$ where $x_i^j \in E$ for any $j \in \{1, \cdots, T\}$. We consider the feature set as a corpus of words and take each sequence of instances as a sentence in the text. Then, we can embed features on one domain into a vector space and calculate correlations among the features.

In this work, we use cosine similarity as the evaluation metric of feature similarities $s(\cdot, \cdot)$ since it is the most common metric used in the word embedding methods to evaluate the similarity between words [17]. After the embedding process, features are represented into low dimensional vectors. Given two features $\mathbf{e_i}$ and $\mathbf{e_j}$, their feature similarity (i.e., cosine similarity) is defined as:

$$s(\mathbf{e_i}, \mathbf{e_j}) = \frac{\mathbf{e_i} \cdot \mathbf{e_j}}{\|\mathbf{e_i}\| \|\mathbf{e_j}\|}, \tag{2}$$

where $\cdot$ represents the inner product in vector space and $\|\cdot\|$ is the Euclidean norm. Note that, given E_s and E_t, we cannot evaluate the similarity between $e_i^s \in E_s$ and $e_j^t \in E_t$ because the two features are represented into different feature spaces. The resulting cosine similarity ranges from -1 to 1, which indicates whether two features are related or not by telling us both the strength and direction of the relationship between features. The closer the similarity are to +1 and -1, the stronger the relationship between the features. As for the relationship direction, $s(\mathbf{u}, \mathbf{v}) > 0$ indicates a positive relationship, while $s(\mathbf{u}, \mathbf{v}) < 0$ indicates a negative relationship. The vectors are independent of each other if $s(\mathbf{u}, \mathbf{v}) = 0$.

In many practical cases, using only positive or negative relationships is far from enough to describe the correlation among features. Instead, we use the Frobenius norm of matrix as the loss function in Equation 1. However, it is unnecessary to give too much attention on exact similarities between feature correlation matrices. Recall the example illustrated in Section 3. The relationship between e_1^s and e_2^s is slightly different from that between e_1^s and e_3^s because $s(e_1^s, e_2^s) = 0.8$ and $s(e_1^s, e_3^s) = 0.9$. But both $s(e_1^s, e_2^s)$ and $s(e_1^s, e_3^s)$ are pretty close to +1 in terms of the strength of $s(\cdot, \cdot)$. Given that the difference between Pearson's correlation coefficients and cosine similarity are negligible [9], we verbally describe the strength of the cosine similarity as the guidance that Evans [10] suggested: strong, moderate, weak and independent. Specifically, we divide the similarity among features into 7 categories: strong positive(3), moderate positive(2), weak positive(1), independent(0), weak negative(-1), moderate negative(-2) and strong positive(-3), which can simplify the problem of capturing the information of feature correlations. The function $\Gamma[(e_i, e_j)]$ that assigns the type of feature similarity between e_i and e_j is defined as:

$$\Gamma[(e_i, e_j)] = \begin{cases} 3, & s(e_i, e_j) > \kappa_s, \\ 2, & \kappa_m < s(e_i, e_j) \le \kappa_s, \\ 1, & \kappa_w < s(e_i, e_j) \le \kappa_m, \\ 0, & -\kappa_w \le s(e_i, e_j) \le \kappa_w, \\ -1, & -\kappa_m \le s(e_i, e_j) < -\kappa_w, \\ -2, & -\kappa_s \le s(e_i, e_j) < -\kappa_m, \\ -3, & s(e_i, e_j) < -\kappa_s, \end{cases} \tag{3}$$

where $\kappa_s > \kappa_m > \kappa_w > 0$. We will explain the choice of κ_s, κ_m and κ_w in details later.

4.2 Feature Isomorphism Discovery

In this section, we describe the key step of TLFid that learns the mapping function among features in different spaces based on the feature isomorphism. Note that in Equation 1, the feature mapping task looks like a popular and effective step, Non-negative Matrix Tri-factorization (NMTF), which tries to learn the mapping function π by decomposing the feature correlation matrix $C(\widetilde{E_t})$ into three submatrices. However, Equation 1 is essentially different from NMTF. Given $C(\widetilde{E_s})$ and $C(\widetilde{E_t})$, TLFid aims to learn π by solve the permutation matrix P_π. The permutation matrix is quite different from the three submatrices in NMTF. It is a square binary matrix that has exactly one entry of 1 in each row and each column with zeros elsewhere. Before describing how to absorb experience from *Graph Isomorphism* to solve this problem, we first give some definitions and background knowledge [3].

Definition 4.1. A feature graph is a 4-tuple $\mathcal{G} = (\mathcal{E}, C, \Phi, \Gamma)$, where

- $\mathcal{E}$ is a finite set of nodes, which represent features in a specific domain
- $C \subset \mathcal{E} \times \mathcal{E}$ is the set of edges between features
- Φ is the label function that assigns labels to features
- Γ is the label function that assigns labels to edges

Note that, in the feature graph, an edge exists between e_i and e_j if and only if $\Gamma[(e_i, e_j)] \neq 0$. Therefore, we transform the feature correlation matrix into a feature graph where each node represents a feature and the label of edge (e_i, e_j) is defined as the type of the feature correlation between e_i and e_j. As for Φ, there can be many options to choose how to assign labels of features. For instance, if we consider each word as one feature in the NLP tasks, Φ can assign semantic meanings or the Part-of-Speech tagging to each word. Notice that both extra labeled information of features and the feature correlation can be incorporated in the TLFid framework.

Definition 4.2. Let $\mathcal{G} = (\mathcal{E}, C, \Phi, \Gamma)$ and $\mathcal{G}' = (\mathcal{E}', C', \Phi', \Gamma')$, be two feature graphs; $\mathcal{G}'$ is a feature subgraph of $\mathcal{G}$, where $\mathcal{G}' \subset \mathcal{G}$, if

- $\mathcal{E}' \subset \mathcal{E}$
- $C' = C \cap \mathcal{E}' \times \mathcal{E}'$
- $\Phi'(e) = \Phi(e)$ for all $e \in \mathcal{E}'$
- $\Gamma'[(e_i, e_j)] = \Gamma[(e_i, e_j)]$ for all $(e_i, e_j) \in C'$

Moreover, for $\widetilde{E_s}$ and $\widetilde{E_t}$ defined in Equation 1, we aim to find the similar subsets over different feature sets. That is, given a graph $\mathcal{G} = (\mathcal{E}, C, \Phi, \Gamma)$, it is prone to know that any subset $\mathcal{E}' \subset \mathcal{E}$ uniquely defines a feature subgraph.

Definition 4.3. Let $\mathcal{G}$ and $\mathcal{G}'$ be two feature graphs. A feature isomorphism between $\mathcal{G}$ and $\mathcal{G}'$ is a bijective mapping function $\pi : \mathcal{E} \rightarrow \mathcal{E}'$, if

- $\Phi(e) = \Phi'(\pi(e))$ for all $e \in \mathcal{E}$
- for any edge $(e_i, e_j) \in C$ there exists am edge $(\pi(e_i), \pi(e_j)) \in C'$ such that $\Gamma[(e_i, e_j)] = \Gamma'[(\pi(e_i), \pi(e_j))]$, and also for any edge $(\pi(e_i), \pi(e_j))$ there exists an edge $(\pi^{-1}(e_i), \pi^{-1}(e_j)) \in C$ such that $\Gamma[(e_i, e_j)] = \Gamma'[(\pi(e_i), \pi(e_j))]$

Definition 4.4. Given a feature isomorphism π from $\mathcal{G}_1$ to $\mathcal{G}_2$, and a subgraph $\mathcal{G}_1$ of another feature graph $\mathcal{G}$, π is called a subgraph isomorphism from $\mathcal{G}_2$ to $\mathcal{G}$.

Many existing transfer learning works are based on the labeled instances to learn a feature mapping function. The performance of these solutions is inferior when labeled instances are not representative. Similarly, we hope that the feature mapping function focuses on the significant features which have the strongest correlations with other features in their own domains. In this paper, given a feature graph $\mathcal{G}_s$ in the source domain, we are interested in: **efficiently querying the subgraph isomorphism π from $\mathcal{G}_s$ to**

$\mathcal{G}_t$, where $\mathcal{G}'_t$ is a subgraph of the feature graph $\mathcal{G}_t$ in the target domain and π is the feature isomorphism from $\mathcal{G}'_t$ to $\mathcal{G}_s$.

So far we have transformed the optimization problem in Equation 1 into a subgraph isomorphism problem, which is proved to be a NP-Complete problem [4]. Fortunately, there exits several excellent works to solve the subgraph isomorphism problem. Considering that one domain may have many features (i.e., $|\mathcal{E}|$ is large), we employ VF2 [5], which is efficient to deal with subgraph isomorphism in large graphs. Due to the space limit, we refer the readers to the original paper [5] for the details of VF2.

4.3 Knowledge Transfer with Feature Mapping Function

In this subsection, we describe how to transfer knowledge from the source domain to the target domain based on the computed bijective feature mapping function π.

Previously, the mapping function π can only perform a one-to-one link between the features that are included in the feature subgraph isomorphism across domains. However, as for the other features, they may also have some useful information even when they are not contained in the feature subgraph isomorphism. In the presence of various feature spaces, it is natural that some structures of representative features are not shared with other domains. Abandoning those features may result in a negative effect on the final performance. Thus, we explain how to preserve such information with the help of π. Given the source domain $\mathcal{S}$ and the target domain $\mathcal{T}$, we have extracted feature sets E_s and E_t from two domains respectively and represent each feature into a latent vector such that $e_i \subset R^{n_s}$ for $e_i \in E_s$ and $e_j \subset R^{n_t}$ for $e_j \in E_t$, where n_s and n_t are the dimensions of vector spaces for the source and the target domains respectively. Suppose that, based on the feature graph isomorphism, we have constructed a feature mapping function $\pi : \bar{E}_s \rightarrow \bar{E}_t$. It is easy to understand that we can utilize $\bar{E}_t$ to represent $\bar{E}_s$. However, mapping the remaining features in the source domain that are not contained in $\bar{E}_s$ is more complex due to the difference in terms of the feature distribution across domains. We define those features as the complementary features E_s^c. We solve the problem by optimizing the loss function with a translation function as follows:

$$\min_{f_s} \mathcal{L}(f_s) = \sum_{e_i^s \in \bar{E}_s} \sum_{e_j^s \in E_s^c} \left| s(e_i^s, e_j^s) - s(\pi(e_i^s), f_s(e_j^s)) \right|^2, \quad (4)$$

where $s(\cdot, \cdot)$ denotes the cosine similarity as defined in Equation 2. Equation 4 aims to preserve the original feature similarities in the source domain and minimize the similarity cost generated from translating the source features to the target feature space. Generally, manifold learning has proposed to identify the low dimensional manifold structure of the given instances and preserve structures in a low dimensional space, such as Isomap [25], locally linear embeddings [23] and Hessian eigenmapping [7]. In this work, we employ one popular manifold learning method, t-distributed Stochastic Neighbor Embedding (t-SNE) which is widely used for visualizing high-dimensional data [16]. While t-SNE focuses on translating features from one domain to another and preserving the original relations among features, we extend it accordingly to perform knowledge transformation in the TLFid framework, the key steps of which are summarized in Algorithm 1.

Algorithm 1 Transfer Learning via Feature Isomorphism Discovery

```
Input: Source data $\mathcal{S} = \{(x_i^s, y_i^s)\}_{i=1}^{\ell_s}$ and target data $\mathcal{T} = \{x_i^t\}_{i=1}^{\ell_t}$.
Output: A classifier $f_*$ that is trained on $\mathcal{S}$, but well performs on $\mathcal{T}$.
1:  procedure EXTRACTING FEATURE CORRELATION MATRIX
2:      Extract features $E_s$ and $E_t$ from source and target domains, respectively.
3:      Apply word embedding to calculate the feature correlation matrices $C(E_s)$ and $C(E_t)$ respectively.
4:  end procedure
5:  procedure FEATURE ISOMORPHISM DISCOVERY
6:      Transform feature spaces on the source and target domain into two feature graphs respectively.
7:      Employ VF2 algorithm to find the feature mapping function $\pi$.
8:  end procedure
9:  procedure KNOWLEDGE TRANSFER WITH FEATURE MAPPING FUNCTION
10:     Apply manifold learning to construct a translator which represents source features into the target feature space. And train a classifier $f_*$ on $\mathcal{S}$ which performs well on $\mathcal{T}$.
11: end procedure
```

5 EXPERIMENTS

In this section, we conduct two experiments to evaluate the effectiveness of the TLFid framework. In the first one, we perform and evaluate the convergence of TLFid with different settings and parameter values. The second experiment compares TLFid with other state-of-the-art transfer learning approaches. All the experiments are performed using the cross-lingual tasks.

5.1 Data Description

In this work, we conduct cross-lingual sentiment classification, to evaluate the performance of the TLFid framework on two benchmark datasets. First multi-lingual sentiment dataset [20] is written in the four languages, English (EN), French (FR), German (GE) and Japanese (JP), which contains reviews on three type of amazon products: books (b), music (m) and DVDs (d). The reviews in each language are split into three categories that 2000 reviews on training set, 2000 reviews on test set and a unlabeled set which varies from 9,000 to 170,000 reviews. We additionally employ one Chinese emotion corpus (R), RenCECps 1.0 [21], as the second sentiment dataset. In terms of grammar and components, Chinese is quite different from other phonography languages such as English, French and German. Thus, we utilize Chinese corpus as the target domain due to very low correspondence between ideograph with phonography. RenCECps contains weblog posts with annotations of sentiment at document, paragraph, and sentence levels. This dataset consists of 1487 documents, with 11,255 paragraphs, 35,096 sentences which annotated for 3-way polarity: positive, neutral and negative. We discard the data labeled as neutral since their polarity is ambiguous.

In this paper, we consider that the reviews of each product in one language are in a feature space. For the comprehensive comparisons, we construct 12 cross-lingual sentiment classification tasks: EN_b-R, EN_m-R, EN_d-R, FR_b-R, FR_m-R, FR_d-R, GE_b-R, GE_m-R, GE_d-R, JP_b-R, JP_m-R and JP_d-R. For example, the task EN_m-R utilizes the music reviews in English as the source domain and the Chinese corpus in the RenCECps as the target domain data.

5.2 Data Preprocessing and Experimental Settings

In this section, we explain some preprocessing works and analyze how model settings affect the performance of the TLFid framework in details. For convenience, we summarize the TLFid framework as shown in Algorithm 1. In the first procedure, before extracting feature correlation matrix, we utilize the TF-IDF to obtain 4987 most meaningful Chinese keywords from RenCECps. There exists two ways to achieve embedding, CBOW and skip-gram. We here use CBOW for embedding due to less running time and employ the word2vec package [17] to represent words into vector spaces and initiate the feature correlation matrices.

As discussed before, in the second step, TLFid assigns different labels to features to boost the accuracy of feature isormorphism. In this paper, we use Google Translator to annotate the Part-of-Speech (POS) in each word. In other words, we use the POS of words as the labels of features such as $\Phi(\text{“}people\text{”}) = \text{“}noun\text{”}$. Moreover, TLFid assigns different labels to the feature correlations as shown in the Equation 3. We set different groups of thresholds $\kappa = (\kappa_s, \kappa_m, \kappa_w)$ to evaluate the accuracy of the TLFid framework. In addition, for simplifying model, we do not set edge between the features e_i with e_j in the feature graph when $-\kappa_w \leq s(e_i, e_j) \leq \kappa_w$.

In the third step of the TLFid framework, since some representative features may not be contained in the maximum feature isomorphism, we represent the complementary features E_s^c based on Equation 4. To verify that the complementary features are useful in the knowledge transfer, we also design an experiment to compare the performance of TLFid with Simple-TLFid which only has first two steps of the TLFid.

5.3 Baseline Methods

After representing features by word embedding, every instance on both of source and target domains is represented with 2-dimensions array. Thus, we configure a 3 layers Long Short-term memory (LSTM) as the binary classifier. In this paper, we mainly compare our model with other algorithms in two situations of the target domain: no labeled instances in $\mathcal{T}$ and scarce labels.

5.3.1 Scarce-Label Baselines. In this paper, we compare TLFid with the following four baselines for the scenarios that scarce labels exist in the source domain:

- **LSTM:** It only utilizes labeled instances on the target domain to train a LSTM binary classifiers [13] where word embedding is applied to preprocess the text data.
- **LIBSVM:** Given that LSTM may not work well on the small dataset, we also employ another robust classifier, LIBSVM [1] with linear kernels and default parameter settings. LIBSVM only utilizes labeled instances on the target domain to train a binary classifiers.
- **DAMA_l:** In this experiment, we define DAMA_l with the same setting in the original work [14]. It aligns domains into a latent space using label information between two domains.
- **HFA:** In this paper, we also employ the HFA [8] which was designed to learn an augmented feature space only based on the labeled examples.
- **SSMC:** SSMC [27] was proposed to explore the correspondence between two languages by constructing a dual-language document-term matrix. We build the semi-supervised matrix for the corpus in the language on the source domain and Chinese.
- **DCI:** DCI [11] aims to build term correspondence by deriving term representations in the vector spaces.

5.3.2 Non-Label Baselines. In the situation that no labeled data exists in the target domain, our proposed method TLFid is compared with the following four baselines:

- **DAMA_f:** In this experiment, we adjust the DAMA's input from data instances to feature representations, E_s and E_t. Also we define the matrix which represents the binary correspondence in DAMA_f between features according to the feature correlation we used in TLFid.
- **SSMC:** In the non-label experiments, we simplify the joint optimization function presented in SSMC [27] by removing prediction loss.
- **HeMap:** HeMap [24] projects data in two domains onto latent space. In this case, we also take the feature representations E_s and E_t as the input instead of data instances. However, it cannot utilize any label information on target.
- **HHTL:** HHTL [28] proposed a deep-learning based method to learn a feature mapping cross domains which can be effective based on the biased corresponding instances. Specifically, we replace corresponding instances with corresponding words where Google translator built the correspondence between cross-lingual words.

5.4 Main Results

Since the target data is balanced, we measure the classification accuracy over all domains.

5.4.1 Evaluation on the TLFid Settings. As discussed in Equation 3, we divide the correlations among feature u with v into 7 categories, strong positive ($s(e_i, e_j) > \kappa_s$), moderate positive ($\kappa_m < s(e_i, e_j) \leq \kappa_s$), weak positive ($\kappa_w < s(e_i, e_j) \leq \kappa_m$), independent ($-\kappa_w \leq s(e_i, e_j) \leq \kappa_w$), weak negative ($-\kappa_m \leq s(e_i, e_j) < -\kappa_w$), moderate negative ($-\kappa_s \leq s(e_i, e_j) < -\kappa_m$) and strong negative ($s(e_i, e_j) < -\kappa_s$). In this subsection, we analyze how correlation threshold κ influences the performance of TLFid (Figure 1). We set four groups of threshold $\kappa = (\kappa_s, \kappa_m, \kappa_w)$: $kappa_1 = (0.7, 0.5, 0.3)$, $kappa_2 = (0.6, 0.4, 0.2)$, $kappa_3 = (0.5, 0.3, 0.1)$ and $kappa_4 = (0.45, 0.25, 0.05)$. Note that we do not set edges between independent features in the feature graph for simplifying the model. Moreover, we also compare TLFid with three simplified TLFid frameworks as follows:

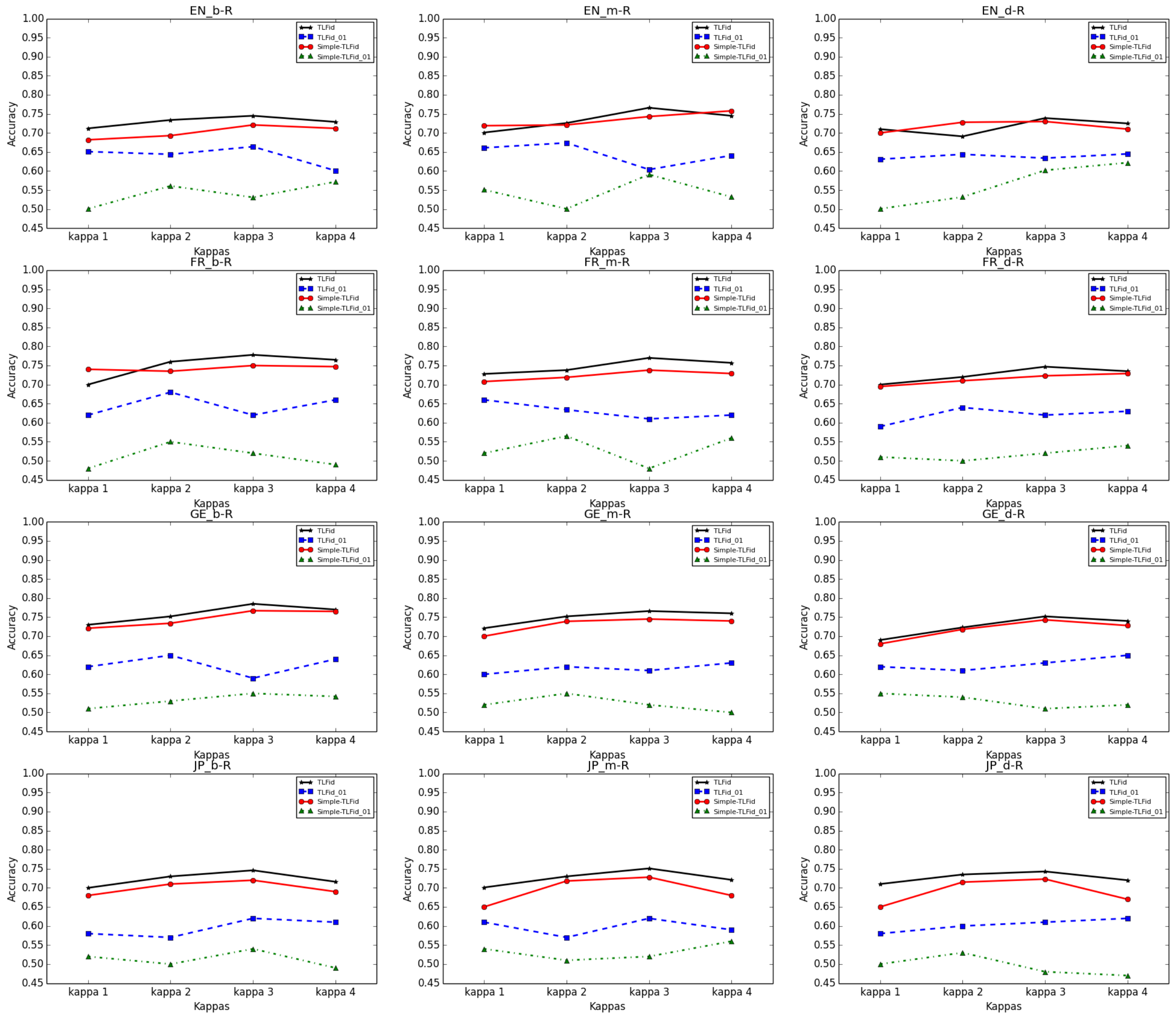

Figure 1: Classification accuracy of TLFid with four groups of κ: $kappa_1 = (0.7, 0.5, 0.3)$, $kappa_2 = (0.6, 0.4, 0.2)$, $kappa_3 = (0.5, 0.3, 0.1)$ and $kappa_4 = (0.45, 0.25, 0.05)$.

- $TLFid_{01}$: In the $TLFid_{01}$ method, we take almost the same step of the original TLFid except the label function of feature correlation Γ. We only assign 3 labels to feature correlations in $TLFid_{01}$, positive ($\kappa_w < s(e_i, e_j)$), independent ($-\kappa_w \leq s(e_i, e_j) \leq \kappa_w$) and negative ($\kappa_w < s(e_i, e_j)$).
- $Simple - TLFid$: In this method, we take the same first two step of the original TLFid. But $Simple - TLFid$ does not represent E_s^c in the target domain and only train the classifier based on those features in the $\bar{E}_s$.
- $Simple - TLFid_{01}$: Based on the $Simple - TLFid$ framework, $Simple - TLFid_{01}$ only considers three categories of feature correlation as the same settings in the $TLFid_{01}$.

As shown in Figure 1, we study the influence of different TLFid frameworks to the overall performance on 12 knowledge transfer tasks. The original TLFid performs more robust and much better than other 3 simple TLFid frameworks. This is because that the TLFid framework embed more knowledge of features on the both domains. Not only TLFid can transfer features that are not contained in the $\bar{E}_s$, but it is able to utilize more categories of feature correlations. Moreover, we study the influence of the κ settings to the overall TLFid framework. Nevertheless, we observe that the best κ setting in various TLFid is $(0.5, 0.3, 0.1)$ which achieves most effectiveness in the cross-lingual sentiment classification. We also employ this setting in the next two comparison experiments.

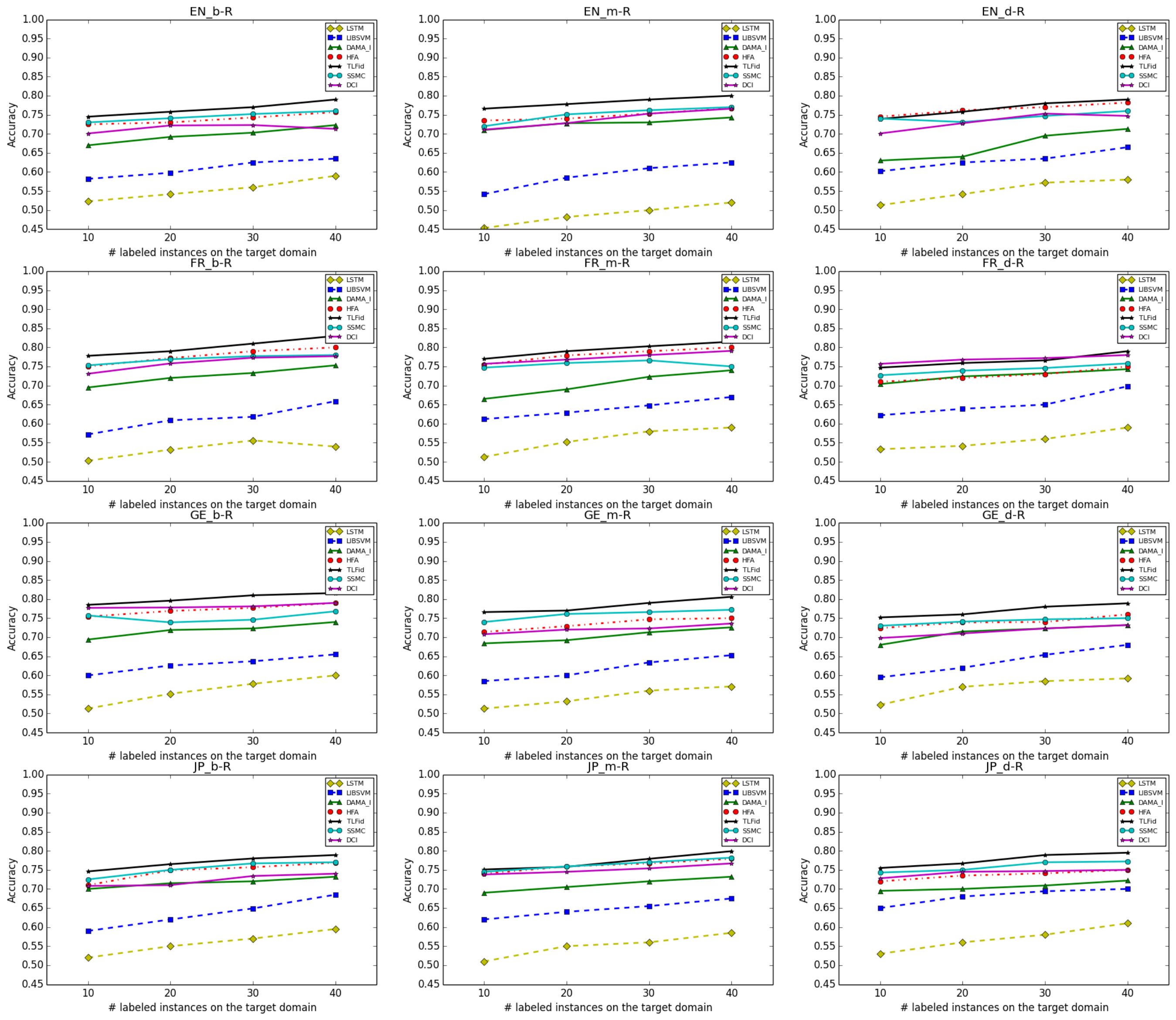

Figure 2: Comparison results in terms of test accuracy (%)

5.4.2 Comparison Results. We test TLFid (with $kappa_3$) and other state-of-art methods on cross-lingual text classification with two situations: 1) **Scarce Labels.** In this experiment, we study the influence of the size of the labeled target instances ([10, 20, 30, 40]) to performance of the TLFid framework. From the results that are reported in Figure 2, the performances increase when using a larger the number of labeled target data. And the two baselines DAMA_l and HFA generally achieve better classification accuracy than LSTM and LIBSVM. Nevertheless, TLFid consistently outperforms other baselines and achieves more stable improvement with increasing size of labeled target data. 2) **No Labels.** To observe the ability to discover or build correspondence for each method, all methods are conducted without any labeled target instances. As shown in Figure 3, generally, all methods outperform HeMap which heavily relies on the strong correspondence among domains. Moreover, our proposed method TLFid outperforms other baselines except for several tasks.

Surprisingly, these results justify the effectiveness of the proposed TLFid framework, which proves that the knowledge from feature correlation benefits the transfer learning task. This is because that, when the low or noisy correspondence exists in two domains, other methods utilize the annotated instances to build new correspondence across domains. The performance of these methods may be unsatisfactory if the labeled instances are not representative in the target domain. In other words, these methods sometimes cannot reveal the true correlation between two domains based on

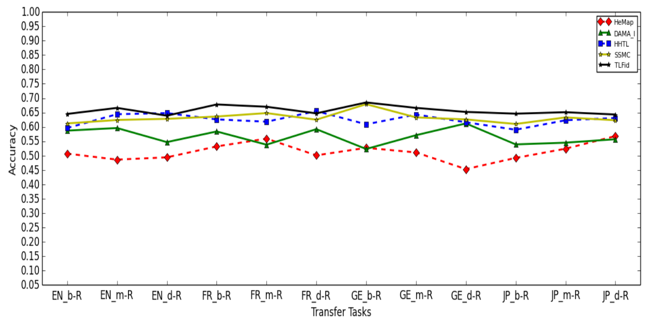

Figure 3: Non-Label cross-lingual sentiment analysis

the biased labeled instances. On the contrary, our proposed method focuses on utilizing knowledge mined from feature correlations among two entire feature spaces, which is not affected by labeled instances. Furthermore, TLFid also benefits from the feature labelling function which has potentials to exploit extra information.

6 CONCLUSION AND FUTURE WORK

In this work, we present a novel transfer learning method TLFid to tackle the heterogeneous transfer learning problem, which utilizes the feature isomorphism across domains and extra feature labeling information to boost the performance of knowledge transfer. We first build the feature mapping function by discovering the feature subgraph isomorphism among different feature spaces. We then exploit the new representations of complementary features by using the mined feature mapping function. We have conducted extensive comparison experiments for two situations of the target domain: unavailable labeled instances and scarce labeled instances. The comparison results demonstrate that our framework TLFid can improve the performance of the cross-lingual sentiment classification tasks by utilizing the correlation of features.

There are still several issues that have yet to be resolved. One difficulty is how to automatically interpret the correlation coefficients. In this paper, we conduct empirical studies on the κ setting. It is necessary to develop an effective algorithm for coefficient interpretation. Furthermore, TLFid exploits extra information by using the label function Φ. We may replace the feature labeling function with co-occurrence data and extend this work to other transfer learning tasks, such as text-aided image classification. We would like to address these issues in the future work and expect this preliminary work could shed light on new approaches to heterogeneous transfer learning.

ACKNOWLEDGMENTS

We thank anonymous reviewers for their insightful and helpful comments, which improve the paper. This work is partially supported by the Hong Kong RGC GRF Project 16207617, RGC G-HKUST603/14, Hong Kong ITC ITS/391/15FX grant, National Grand Fundamental Research 973 Program of China under Grant 2014CB340303, the National Science Foundation of China (NSFC) under Grant No. (61729201, 61602297), Scienceand Technology Planning Project of Guangdong Province, China, No. 2015B010110006. The corresponding author of this paper is Yanyan Shen.

BIBLIOGRAPHY

[1] Chih-Chung Chang and Chih-Jen Lin. 2011. LIBSVM: a library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)* 2, 3 (2011), 27.

[2] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research* 12, Aug (2011), 2493–2537.

[3] Donatello Conte, Pasquale Foggia, and Mario Vento. 2007. Challenging Complexity of Maximum Common Subgraph Detection Algorithms: A Performance Analysis of Three Algorithms on a Wide Database of Graphs. *J. Graph Algorithms Appl.* 11, 1 (2007), 99–143.

[4] Stephen A Cook. 1971. The complexity of theorem-proving procedures. In *Proceedings of the third annual ACM symposium on Theory of computing*. ACM, 151–158.

[5] Luigi P Cordella, Pasquale Foggia, Carlo Sansone, and Mario Vento. 2004. A (sub) graph isomorphism algorithm for matching large graphs. *IEEE transactions on pattern analysis and machine intelligence* 26, 10 (2004), 1367–1372.

[6] Wenyuan Dai, Yuqiang Chen, Gui-Rong Xue, Qiang Yang, and Yong Yu. 2009. Translated learning: Transfer learning across different feature spaces. In *Advances in neural information processing systems*. 353–360.

[7] David L Donoho and Carrie Grimes. 2003. Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data. *Proceedings of the National Academy of Sciences* 100, 10 (2003), 5591–5596.

[8] Lixin Duan, Dong Xu, and Ivor Tsang. 2012. Learning with augmented features for heterogeneous domain adaptation. *arXiv preprint arXiv:1206.4660* (2012).

[9] Leo Egghe and Loet Leydesdorff. 2009. The relation between Pearson's correlation coefficient r and Salton's cosine measure. *Journal of the Association for Information Science and Technology* 60, 5 (2009), 1027–1036.

[10] James D Evans. 1996. *Straightforward statistics for the behavioral sciences*. Brooks/Cole.

[11] Alejandro Moreo Fernández, Andrea Esuli, and Fabrizio Sebastiani. 2016. Distributional Correspondence Indexing for Cross-Lingual and Cross-Domain Sentiment Classification. *Journal of artificial intelligence research* 55 (2016), 131–163.

[12] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 855–864.

[13] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[14] Judy Hoffman, Erik Rodner, Jeff Donahue, Trevor Darrell, and Kate Saenko. 2013. Efficient learning of domain-invariant image representations. *arXiv preprint arXiv:1301.3224* (2013).

[15] David G Lowe. 2004. Distinctive image features from scale-invariant keypoints. *International journal of computer vision* 60, 2 (2004), 91–110.

[16] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research* 9, Nov (2008), 2579–2605.

[17] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[18] Sinno Jialin Pan, Ivor W Tsang, James T Kwok, and Qiang Yang. 2011. Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks* 22, 2 (2011), 199–210.

[19] Sinno Jialin Pan and Qiang Yang. 2010. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering* 22, 10 (2010), 1345–1359.

[20] Peter Prettenhofer and Benno Stein. 2010. Cross-language text classification using structural correspondence learning. In *Proceedings of the 48th annual meeting of the association for computational linguistics*. Association for Computational Linguistics, 1118–1127.

[21] Changqin Quan and Fuji Ren. 2010. A blog emotion corpus for emotional expression analysis in Chinese. *Computer Speech & Language* 24, 4 (2010), 726–749.

[22] Rajat Raina, Alexis Battle, Honglak Lee, Benjamin Packer, and Andrew Y Ng. 2007. Self-taught learning: transfer learning from unlabeled data. In *Proceedings of the 24th international conference on Machine learning*. ACM, 759–766.

[23] Sam T Roweis and Lawrence K Saul. 2000. Nonlinear dimensionality reduction by locally linear embedding. *science* 290, 5500 (2000), 2323–2326.

[24] Xiaoxiao Shi, Qi Liu, Wei Fan, and S Yu Philip. 2013. Transfer across completely different feature spaces via spectral embedding. *IEEE Transactions on Knowledge and Data Engineering* 25, 4 (2013), 906–918.

[25] Joshua B Tenenbaum, Vin De Silva, and John C Langford. 2000. A global geometric framework for nonlinear dimensionality reduction. *science* 290, 5500 (2000), 2319–2323.

[26] Chang Wang and Sridhar Mahadevan. 2011. Heterogeneous domain adaptation using manifold alignment. In *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, Vol. 22. 1541.

[27] Min Xiao and Yuhong Guo. 2014. Semi-Supervised Matrix Completion for Cross-Lingual Text Classification.. In *AAAI*. 1607–1614.

[28] Joey Tianyi Zhou, Sinno Jialin Pan, Ivor W Tsang, and Yan Yan. 2014. Hybrid Heterogeneous Transfer Learning through Deep Learning.. In *AAAI*. 2213–2220.