Feature Selection via Transferring Knowledge Across Different Classes

ZHENG WANG, University of Science and Technology Beijing, Tsinghua University
XIAOJUN YE, Tsinghua University
CHAOKUN WANG, Tsinghua University
PHILIP S. YU, University of Illinois at Chicago

The problem of feature selection has attracted considerable research interest in recent years. Supervised information is capable of significantly improving the quality of selected features. However, existing supervised feature selection methods all require that classes in the labeled data (source domain) and unlabeled data (target domain) to be identical, which may be too restrictive in many cases. In this paper, we consider a more challenging cross-class setting where the classes in these two domains are related but different, which has rarely been studied before. We propose a Cross-class Knowledge Transfer Feature Selection (CKTFS) framework which transfers the cross-class knowledge from the source domain to guide target domain feature selection. Specifically, high-level descriptions, i.e., attributes, are used as the bridge for knowledge transfer. To further improve the quality of the selected features, our framework jointly considers the tasks of cross-class knowledge transfer and feature selection. Experimental results on four benchmark datasets demonstrate the superiority of the proposed method.

CCS Concepts: • **Information systems → Data mining**; • **Computing methodologies → Feature selection**.

Additional Key Words and Phrases: Feature selection, dimension reduction, supervision transfer.

ACM Reference Format:
ZHENG WANG, XIAOJUN YE, CHAOKUN WANG, and PHILIP S. YU. 2019. Feature Selection via Transferring Knowledge Across Different Classes. *ACM Trans. Knowl. Discov. Data.* 1, 1, Article 1 (January 2019), 29 pages. https://doi.org/10.1145/3314202

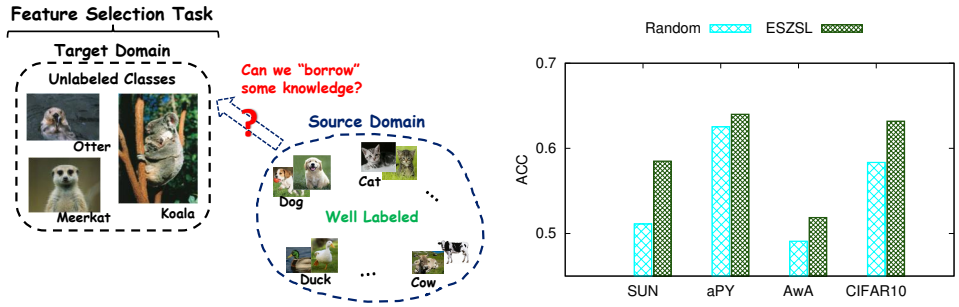
1 INTRODUCTION

In many real-world applications such as multimedia processing and computer vision, data is usually represented by high-dimensional features [80] [14]. In practice, high-dimensional data often contains lots of redundancy and noise, which will result in high computational cost, huge storage requirements, or even overfitted models [46] [79]. To alleviate this, dimension reduction techniques are introduced to transform high-dimensional data into a lower dimensional space while attempting to preserve the characteristics of the original data.

This work is supported in part by China Postdoctoral Science Foundation Funded Project (No. 2018M640066), Fundamental Research Funds for the Central Universities (No. FRF-TP-18-016A1), and National Natural Science Foundation of China (No. 61872207).

Authors' addresses: Z. Wang is with Department of Computer Science and Technology, University of Science and Technology Beijing, Beijing, China, Email: wangzheng@ustb.edu.cn; X. Ye, and C. Wang are with School of Software, Tsinghua University, Email: yexj@mail.tsinghua.edu.cn and chaokun@mail.tsinghua.edu.cn; P.S. Yu is with Department of Computer Science, University of Illinois at Chicago, Email: psyu@uic.edu. Chaokun Wang is the corresponding author.

1556-4681/2019/1-ART1 $15.00
https://doi.org/10.1145/3314202



(a) Feature selection in a "cross-class" setting (b) Feature selection performance based on ZSL

Fig. 1. The problem of feature selection in a "cross-class" setting and it's possible solution based on ZSL. (a) In this setting, the labeled source domain and unlabeled target domain have related but totally different classes; (b) The average feature selection performance of Random and ESZSL methods (feature number ranges in [50,100, ..., 500]).

The existing dimension reduction techniques can roughly be sub-divided into two categories: feature extraction and feature selection. Feature extraction [38] [45] [7] generates new features by projecting the original higher-dimensional feature space to a lower-dimensional feature space. Feature selection [24] [62] [78] takes an alternative approach to dimension reduction by locating a few most relevant features, rather than producing an entirely new set of dimensions. Compared to feature extraction, feature selection maintains physical meaning of the original features and has better interpretability. In this study, therefore, we focus on feature selection.

Label information has shown its effectiveness for discriminative feature selection [6] [48]. Nevertheless, supervised feature selection methods often require a large amount of labeled data which is often costly to obtain. To overcome the scarcity of labeled data, two lines have been widely exploited. The first line is semi-supervised learning [81] [48] which uses both labeled data (i.e., source domain) and unlabeled data (i.e., target domain) to estimate feature importance. The second line is transfer learning [56] assuming there exists a well-labeled auxiliary source domain whose data distribution is related to but not the same as the target domain. By reducing the distribution difference between these two domains, supervised knowledge is transferred from the auxiliary source domain to guide target domain feature selection [21] [26].

However, all these above-mentioned methods require the classes in the source and target domains to be identical, while little attention has been paid to a more challenging case where these two domains have related but different classes, i.e., **cross-class case**. This case deserves special attention for two reasons. Firstly, it has many practical applications. As illustrated in Fig. 1(a), in some uncommon categories such as "otter" and "koala", it is very difficult to collect labeled data for all these categories exactly and not miss any category. On the other hand, collecting labeled training data in some other more common categories, like "cat" and "dog", is much easier. This motivates us to consider whether we can "borrow" some supervised knowledge from these well-labeled common categories to guide the feature selection in those uncommon categories where little label information exists. Secondly, and more importantly, traditional feature selection methods may perform poorly in the target domain, due to the lack of supervised information. Moreover, some supervised methods might even be misled by the supervision information of source domain, due to the big difference between the source domain and target domain.

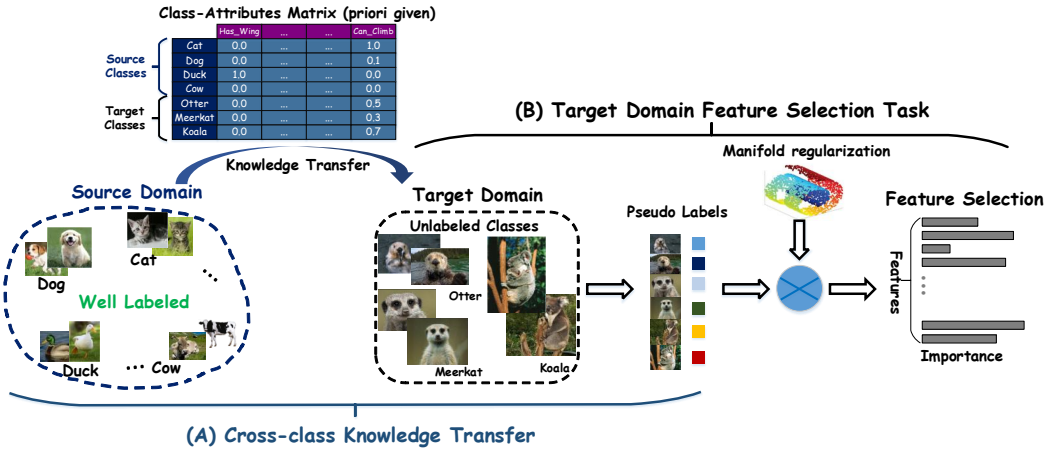


Fig. 2. Illustration of Cross-class Knowledge Transfer Feature Selection: (A) Cross-class Knowledge Transfer; (B) Target Domain Feature Selection. The main idea behind our approach is that the target domain feature selection (i.e., part B) should benefit from the cross-class knowledge transfer (i.e., part A).

The major challenge in this problem is how to borrow some knowledge, which is stable across different classes, from the labeled source domain to the unlabeled target domain. This is also the primary reason why existing feature selection methods fail to handle the studied problem. Recently, the idea of Zero-Shot Learning (ZSL) [35] [20], which adopts attributes as an bridge for cross-class knowledge transfer, has been explored to recognize unseen new classes. This seems to be a straightforward solution to our problem. To verify this consideration, we conduct an experiment on four widely used datasets which have attributes and source/target split. Firstly, we use ESZSL [60], a well-known ZSL method, to infer the "pseudo labels" of target instances. After that, we feed these pseudo labels to the classical supervised feature selection method LASSO [70], so as to perform feature selection on target data. Finally, we evaluate the feature selection performance via K-means clustering. Figure 1(b) shows the clustering performance of ESZSL and Random (i.e., selecting features randomly), in item of Clustering Accuracy (ACC) [17]. We can evidently conclude the usefulness of these pseudo labels, although the classification of ESZSL is actually very inaccurate (it's classification accuracies on these four datasets range in [15% $\sim$ 69%], and more details can be found in Section 7).

An attendant problem is how to select more discriminative features with the cross-class transferred knowledge which might be quite noisy and inaccurate [35]. To address this, as illustrated in Fig. 2, we further force these pseudo labels to fit the manifold structure of target data, so as to guide feature selection in this domain. As such, our method not only utilizes the supervised knowledge of source data, but also captures the intrinsic structure of target data. Intuitively, by adopting attributes as well as pseudo labels, we connect the tasks of cross-class knowledge transfer and feature selection. In addition, we jointly consider these two tasks in a unified framework for more effective and accurate learning. Finally, we derive an effective algorithm to solve the optimization problem of the proposed framework.

We evaluate the performance of our method on several real-world benchmark datasets. One point should be noted is that the class-attribute descriptions of a dataset named CIFAR10 are automatically generated from a public Wikipedia text-corpus [65] by a well-known NLP tool [29].

The experimental results show that no matter with manually or automatically annotated attributes, our method can be an effective way to "borrow" cross-class knowledge to promote feature selection.

We highlight the contributions of this paper as follows.

- We study a challenging and practical problem: feature selection when the source and target domains have related but totally different classes. To our best knowledge, this is the first attempt to enhance feature selection by transferring knowledge from other different classes.
- We propose a novel feature selection framework CKTFS, which jointly considers the tasks of cross-class knowledge transfer and target domain feature selection. An efficient algorithm is further proposed for the optimization problem of CKTFS.
- We conduct extensive experiments on four benchmark datasets to demonstrate the effectiveness of the proposed method.

The remainder of this paper is organized as follows. Section 2 reviews some related work. In Section 3, we elaborate our framework with details. To solve the optimization issue in the proposed framework, an an effective optimization approach together with its time complexity analysis is given in Section 4. In Section 5, we discuss possible extensions of our framework. Section 6 analyses the relationship between some existing feature selection methods and ours. Section 7 reports experimental results. Section 8 concludes this paper with future work.

2 RELATED WORK

In this section, we briefly review three related lines of research, and highlight the difference between these works and ours.

2.1 Feature Selection

Suppose the whole dataset contains a fully labeled part (source domain) and an unlabeled part (target domain). With regard to how to use these two domains, feature selection methods fall into three groups: unsupervised, supervised, and semi-supervised feature selection methods. When the labeled source domain is not provided, feature selection methods are unsupervised. They select features which best keep the intrinsic structure of target data according to various criteria, such as data variance [9], data similarity [25] [53] and data separability [5] [42]. To further select more discriminative features, some unsupervised methods [28] [43] propose to learn pseudo labels (i.e., cluster indicators) and perform feature selection simultaneously.

Supervised feature selection methods seek features that are efficient for discrimination in the labeled source domain. Classical methods, such as Pearson Correlation Coefficient [40] and Fisher Score [16], evaluate features' weights according to labels and select features one by one. To further exploit the correlation among features [67], lots of sparsity-based feature selection methods are proposed. One of the most famous method is LASSO [70] which selects features by constraining the ℓ_1-norm of weights. Besides, another sparsity regularization model $\ell_{2,1}$-norm has gained increasing interest for its sparsity, joint selection way and ability to exploit the pairwise correlation among features. For example, [52] adds $\ell_{2,1}$-norm on both learned regression model and regularization term (for feature selection). [84] embeds classifier learning with subspace learning, and adds $\ell_{2,1}$-norm regularization term (for feature selection).

Extended from unsupervised and supervised feature selection methods, semi-supervised methods consider both source and target domains for feature selection. For example, [48] fully explores the distribution of the labeled and unlabeled data with a label propagation method, and then selects top features based on the noise insensitive trace ratio criterion. [74] proposes a semi-supervised multi-label feature selection method which jointly utilizes the label correlations and exploits data structure by manifold learning.

All the above mentioned methods do not consider the case where the source domain and target domain have totally different classes. In this paper, we propose to transfer cross-class knowledge from source domain to generate pseudo labels for target domain feature selection. Note that, although some unsupervised methods [28] [43] also propose to learn pseudo labels, their methods are substantially different from ours. Specifically, in these methods, pseudo labels are learned by exploiting the cluster analysis on the target data, i.e., they do not consider the knowledge of source domain. Conversely, in our method, pseudo labels are generated, via attributes as the bridge, by transferring knowledge from the source domain to target domain. In addition, we further force the generated pseudo labels to preserve the manifold structure of target data, so as to guide feature selection in this domain.

2.2 Transfer Learning for Feature Selection

Transfer learning, also known as learning to learn [36], or inductive transfer [8] [58], has the goal to transfer the knowledge learned in previous source domain to the target domain whose data distribution is similar but different from source domain. One of the most important transfer strategies is the feature-representation-based-transfer approach. Its basic idea is to employ feature extraction or selection methods to learn a "good" feature representation to reduce the distribution difference between source and target domains. For example, [10] and [55] reduce this difference by learning a common feature extraction strategy. Uguroglu et al. [71] reduce this difference by selecting distribution invariant features across domains.

To overcome the scarcity of labeled data, some feature selection methods also consider transfer learning. Bi et al. [2] compute "confidence" scores for source domain instances according to their feature distribution difference with target ones, and use high-confidence ones to enrich the target domain's training set for feature selection. In the word sense disambiguation task, Dhillon et al. [13] improve feature selection performance by introducing feature relevance prior inferred from other "similar" word senses. Helleputte et al. [26] further incorporate this prior into an optimization framework for feature selection. Fukumoto et al. [21] consider the impact of temporal effects for transfer learning based feature selection.

However, these transfer learning based feature selection methods still require that classes in source and target domains to be identical, i.e., they cannot transfer knowledge from other different classes for feature selection. Besides, unlike these transfer learning based feature selection methods which transfer knowledge by reducing the distribution difference between source and target domains, we utilize attributes as a bridge to transfer knowledge from source domain to guide target domain feature selection.

2.3 Attribute-based Zero Shot Learning

Attribute-based Zero Shot Learning (ZSL) [35] [20] shares the same aim with transfer learning, i.e., applying the knowledge learned from source domain to target domain. The main difference is that: in ZSL, the classes in source and target domains are different. In other words, ZSL transfers knowledge across different classes. To achieve this goal, it uses some high-level descriptions shared among the classes in these two domains, i.e., attributes, as the bridge for knowledge transfer. Generally speaking, there are two ways to generate attributes. On the one hand, we can manually describe each class by a list of pre-defined attributes. Take the animals in Fig. 2 as an example, possible attributes may include "has wing", "can climb" or "is black". On the other hand, since class labels are usually semantic meaningful (like "cat" and "dog"), we can automatically generate their semantic embedding vectors as attributes. More concretely, we can first collect some linguistic resources (like Wikipedia corpus). Then we can use some word embedding NLP tools (such as word2vec [51] or Huang's method [29]) to embed all the words (including those classes labels)

into a latent space, where the proximity between points in space indicates semantic proximity. Intuitively, in this embedded space, each dimension can be seen a "latent" attribute which reflects the semantic relationship among different classes.

Recently, various ZSL methods have been proposed, such as attribute-based methods [35] [30] and similarity-based methods [59] [54]. However, all these methods are limited to classification or prediction scenario. To our best knowledge, this is the first attempt to utilize the knowledge transferred from other different classes to promote feature selection.

3 THE PROPOSED METHOD

In this section, we first introduce some notations used throughout this paper. Then, we describe the formulation of our Cross-class Knowledge Transfer Feature Selection (CKTFS) framework. After that, we provide an effective solution to address the optimization issue of CKTFS. Finally, we analyse the computational complexity of the optimization method.

3.1 Problem Definition and Discussion

3.1.1 Problem Definition. We have a source domain $\mathcal{D}_s$ and a target domain $\mathcal{D}_t$. In the source domain $\mathcal{D}_s$, there is a set of source instances $X_s=\{x_1^s, ..., x_{n_s}^s\}$, where $x_i^s \in \mathbb{R}^d$ is the feature vector. We assume that all source instances are labeled: the class set is $C_s=\{C_1^s, ..., C_{c_s}^s\}$ and the label information is $Y_s = \{y_1^s, ..., y_{n_s}^s\}$ where $y_i^s \in \{0,1\}^{c_s}$ is the label vector. In addition, in this paper, we only consider the multi-class single-label case, i.e., there is only one "1" in each y_i^s and the others are all "0".

In the target domain $\mathcal{D}_t$, there is a set of target instances $X_t=\{x_1^t, ..., x_{n_t}^t\}$, where $x_i^t \in \mathbb{R}^d$ is the feature vector[1]. The class set of this domain is $C_t=\{C_1^t, ..., C_{c_t}^t\}$, but we do not know its label information $Y_t = \{y_1^t, ..., y_{n_t}^t\}$ where $y_j^t \in \{0,1\}^{c_t}$ is the label vector. Our goal is to select a subset of most discriminative features in this domain. Different from the traditional supervised feature selection setting C_s=C_t, this paper considers a more practical and challenging case (i.e., **cross-class** case) where the source and target classes are disjoint: $C_s \cap C_t=\varnothing$.

In addition, for each class in $C_s \cup C_t$, there are a attributes describing its high-level characteristics. In this way, for domain $\mathcal{D}_s$ and $\mathcal{D}_t$, we can get two Class-Attributes matrices $S_s \in \mathbb{R}^{a,c_s}$ and $S_t \in \mathbb{R}^{a,c_t}$, respectively. These two matrices may contain boolean entries, when classes are described by a combination of attributes, or more generally, they may contain for each attribute any value in [0,1] providing a soft link between attributes and classes. For clarity, we summarize these notations in Table 1.

Finally, having all aforementioned notations, we can formally define the concerned problem studied in this paper as follows:

Problem 1 (Cross-class Feature Selection, CCFS). *Given a labeled source domain $\mathcal{D}_s=(X_s, S_s, Y_s)$ and an unlabeled target domain $\mathcal{D}_t=(X_t, S_t)$ where the classes in $\mathcal{D}_s$ and $\mathcal{D}_t$ are totally different, our goal is to utilize the supervision knowledge of $\mathcal{D}_s$ to improve the feature selection task in $\mathcal{D}_t$.*

3.1.2 Problem Discussion. We discuss the practical significance of the CCFS problem (defined in Problem 1) in which two observations are available. The first observation is that all the classes in the target domain have no labeled instances. This has practical implications for two reasons. Firstly, and actually, there are few labeled instances in most of the real-world categories [18]. This is reflected in the image number per category, available for training in several large object

[1]Here, following the general setting of feature selection and ZSL studies, we assume source and target instances are all in the same feature space $\mathbb{R}^d$. In other words, source and target instances are pre-processed by the same feature extraction tools (e.g., SIFT [49] and AlexNet [34]). This is a common assumption in the literature, because it is convenient to use the same feature extraction methods or tools for data pre-processing [19] [32].

Table 1. NOTATIONS

Symbol	Description
$\mathcal{D}_s$, $\mathcal{D}_t$	source and target domains
X_s, X_t	source and target instances
Y_s, Y_t	label matrix of source and target instances
S_s, S_t	attribute matrix of source and target classes
C_s, C_t	source and target class set
n_s, n_t	#source instances and #target instances
c_s, c_t	#source classes and #target classes
a	#attributes for describing different classes
$h_{(S_s)}$, $h_{(S_t)}$	classifier derived from attribute matrix of source and target classes
h	the derivation function

datasets (like ImageNet [11]), shows a Zipf distribution [63]. Moreover, in light of the knowledge explosion, this situation would become worse with the rapid growth of newly-emerging concepts and multimedia data (like the newly invented product "quadrotor").

The second observation is that the class-attribute descriptions of source and target classes are provided. Actually, this would make our method more applicable in practice. Comparing to collecting labeled data, collecting class-attribute descriptions is easier. On the one hand, labeled data must be provided in instance-level, i.e., people have to annotate a number of instances for each class. Moreover, it is usually costly and difficult to annotate sufficient labeled data in practice [76]. On the other hand, class-attribute descriptions can be provided in class-level, i.e., people only need to provide one attribute description vector for each class. In addition, as pointed out in Section 2.3, this kind of description not only can be annotated manually, but also can be automatically generated. For example, for each class, we can use the word embedding vector of this class's name as its attribute description vector. Actually, lots of studies [66] [12] have shown that some public word embedding datasets (like Huang's project [29] [2]) perform well on a variety of attribute-related tasks. This is also validated in our experiments.

3.2 Cross-Class Feature Selection

3.2.1 Cross-class knowledge transfer. In the source domain $\mathcal{D}_s$, given a set of labeled data (x_i^s, y_i^s), we can learn a multi-class classification model by minimizing a regularized classification loss:

$$\min_f \sum_{i=1}^{n_s} \mathcal{L}(f(x_i^s), y_i^s) + \mathcal{R}(f) \tag{1}$$

where $\mathcal{L}(\cdot,\cdot)$ is a loss function, f is the learned classifier, and $\mathcal{R}$ is the regularization term for f to avoid overfitting. The value of $f(x_i^s)$ can be seen as the compatibility score between the input instance x_i^s and different source classes.

However, as no target instances are labeled for training, Eq. 1 cannot be directly applied for cross-class knowledge transfer. To address this issue, existing ZSL methods [35] [60] introduce attributes for cross-class knowledge transfer. Specifically, considering the fact that a class is well characterized by its attributes, it is reasonable to assume that the class classifier could be derived from the class-attribute descriptions. In other words, we can set $f = h_{(S_s)}$, where $h_{(S_s)}$ is the classifier derived from source class-attribute descriptions and h is the derivation function matching

[2]http://ai.stanford.edu/ ehhuang/

features with attributes. Therefore, we can rewrite Eq. 1 as the following optimization problem:

$$\min_{h} \sum_{i=1}^{n_s} \mathcal{L}(h_{(S_s)}(x_i^s), y_i^s) + \mathcal{R}(h_{(S_s)}) \tag{2}$$

Intuitively, by matching features with class-attribute representations, $h_{(S_s)}(x_i^s)$ measures the compatibility between instance x_i^s and different seen classes. Ideally, the most compatible attribute representation of each training instance should come from its ground-truth class. Therefore, this objective function can fully exploit the discriminability of attribute representations by assigning each instance to the class with the most compatible attribute representation. On the other hand, as attributes are generally shared by both seen and unseen classes, the learned derivation function h should be able to generalize to the unseen classes [60]. Therefore and similarly, we can predict the label vector of a target instance $x^t \in X_t$ by measuring the compatibility between x^t and different target classes as: $y^t = h_{(S_t)}(x^t)$, where $h_{(S_t)}$ denotes the classifier derived from target class-attribute descriptions. In the target domain, we call these predicted labels as *pseudo labels*, because we actually do not know the true labels of target data.

We can incorporate the target domain into the objective function Eq. 2, since our ultimate goal is to select valuable features in this domain. Therefore, we can transductively infer the pseudo labels of target instances (i.e., $Y_t = \{y_1^t, ..., y_{n_t}^t\}$) by minimizing the following classification loss:

$$\begin{aligned} \min_{h, Y_t} \quad & \sum_{i=1}^{n_s} \mathcal{L}(h_{(S_s)}(x_i^s), y_i^s) + \sum_{j=1}^{n_t} \mathcal{L}(h_{(S_t)}(x_i^t), y_j^t) + \mathcal{R}(h_{(S_s)}) + \mathcal{R}(h_{(S_t)}) \\ \text{s.t.} \quad & \|Y_t(j,:)\|_0 = Y_t(j,:)\mathbf{1}_{c_t} = 1, \ j = 1, \ldots, n_t \end{aligned} \tag{3}$$

where $\|\cdot\|_0$ denotes the ℓ_0-norm of a vector, and $\mathbf{1}_{c_t}$ is a column vector with all its elements being 1. The constraint in Eq. 3 is imposed to ensure each target instance only belongs to one target class, so as to be consistent with our multi-class single-label setting. In addition, this constraint guarantees discriminative feature selection, since discriminative information is usually encoded in the form of labels [27].

The intuition behind Eq. 3 is that the classifier is able to learn the relationship between the raw-input features and semantic descriptions (i.e., attributes). When a new unseen instance is presented, the classifier could make a prediction about its attributes. As such, we can recognize this unseen instance by comparing its predicted attributes with the attributes of each unseen concept. As a concrete example, if a testing image is classified as having attributes "stripes", "four legs", and "can climb", it is more likely to be a cat than a bird or fish, even without having seen any images of these three animals during training.

3.2.2 Target domain feature selection. As described above, we have introduced pseudo labels, via attributes as the bridge, into the unlabeled target domain. However, these pseudo labels are learned individually, and thus fail to preserve the intrinsic geometry structure of target data. This would degrade the feature selection performance, especially considering that high-dimensional data often presents a low-dimensional manifold structure [15] [47]. Therefore, to capture this structure, we follow the general idea of manifold learning [44] [82] to ensure similar target data have similar pseudo labels. This yields the following loss term:

$$\frac{1}{2} \sum_{i,j=1}^{n_t} G_{i,j} \left\| y_i^t - y_j^t \right\|_F^2 = Tr(Y_t' L Y_t) \tag{4}$$

where $L = A - G$ is the Laplacian matrix and A is a diagonal matrix with its i-th diagonal entry being the sum of the i-th row of G. G is a weight matrix, whose element G_{ij} reflects the similarity

between two target instance x_i^t and x_j^t as

$$G_{ij} = \begin{cases} 1, & \text{if } x_i^t \in \mathcal{N}_k(x_j^t) \text{ or } x_j^t \in \mathcal{N}_k(x_i^t); \\ 0, & \text{otherwise,} \end{cases}$$

where $\mathcal{N}_k(x^t)$ denotes the k-nearest neighbors of instance x^t, and the similarity of two instances is evaluated by Euclidean distance in this paper. Actually, to preserve structure manifold, other kernels (like Gaussian kernel) could also be adopted to construct the weight matrix G. We will compare different kernel sittings in our experiments.

With the introduced pseudo labels, from a generative point of view, we assume that these pseudo labels could be generated by a small subset of features. This assumption yields the following optimization problem:

$$\min_{g} \sum_{j=1}^{n_t} O(g(x_j^t), y_j^t) + \Omega(g) \tag{5}$$

where $O(\cdot, \cdot)$ is a loss function, g is a generation function, and Ω is the sparse regularization term. The sparse regularization term Ω forces some feature coefficients to be small or exactly zero, and then their corresponding features can be simply eliminated. Intuitively, this objective function seeks to approximate pseudo labels by a sparse superposition of features, making it particularly suitable for feature selection.

The rationality of the above feature selection strategy (i.e., Eq. 4 and Eq. 5) is supported by the manifold learning theory [44] [82], i.e., high-dimensional data often presents a low-dimensional manifold structure. For example, [61] has shown the 2-d manifold of high-dimensional face images (i.e., 20*28 features) could clearly reflect the pose and expression of human faces. In our method, Eq. 4 firstly embeds the original high-dimensional data into a low-dimensional space, so as to retain the most valuable information of original data. After that, Eq. 5 selects a small subset of features to describe this value information, so as to reduce the feature redundancies. Intuitively, our method prefers those features which could describe this low-dimensional manifold structure.

3.2.3 Cross-class knowledge transfer feature selection framework. By jointly considering the cross-class knowledge transfer part (Eq. 3) and target domain feature selection part (Eq. 4 and Eq. 5), we can effectively utilize the transferred knowledge from the source domain to guide the feature selection in the target domain, which is formulated as follows:

$$\begin{aligned} \min_{h, Y_t, g} \ & \mathcal{J}_{\text{CKTFS}} = \sum_{i=1}^{n_s} \mathcal{L}(h_{(S_s)}(x_i^s), y_i^s) + \sum_{j=1}^{n_t} \mathcal{L}(h_{(S_t)}(x_i^t), y_j^t) \\ & + \alpha Tr(Y_t' L Y_t) + \beta \sum_{j=1}^{n_t} O(g(x_j^t), y_j^t) + \lambda(\mathcal{R}(h_{(S_s)}) + \mathcal{R}(h_{(S_t)}) + \Omega(g)) \\ \text{s.t.} \ & \|Y_t(j,:)\|_0 = Y_t(j,:)\mathbf{1}_{c_t} = 1, \ j = 1, \ldots, n_t \end{aligned} \tag{6}$$

where α, β and λ are balance parameters.

The core of the proposed framework is the use of pseudo labels whose significance mainly lies in two-fold. On the one hand, with pseudo labels, we can select discriminative features in a "supervised" manner. On the other hand, besides capturing the manifold structure of the unlabeled target data, the pseudo labels used in our method also benefit from the success of attribute based cross-class knowledge transfer. This is the primary difference between our method and other unsupervised feature selection methods (like JELSR [28] and DisUFS [68]) which also use pseudo labels. Intuitively, by adopting pseudo labels, we successfully connect the cross-class knowledge

transfer task and target domain feature selection task. As such, the performance of target domain feature selection would be significantly improved by the cross-class transferred knowledge.

Note that Eq. 6 is the general framework of the proposed CKTFS, in which we can choose different fitting function h and g, loss function $\mathcal{L}$ and O, regularization $\mathcal{R}$, and sparse regularization Ω. Therefore, users can customize it based on the specific demand.

3.3 A Customized Feature Selection Model Based on CKTFS Framework

In this subsection, we give a simple but effective feature selection method by customizing the two core parts of the proposed framework (Eq. 6). Specifically, for the cross-class knowledge transfer part (Eq. 3), we adopt a linear model for h, i.e., $h_{(S_s)}(x_i^s) = x_i^s W S_s$, where $W \in \mathbb{R}^{d,a}$ is the linear model that matches features with class-attribute representations. Intuitively, WS_s can be seen as the class-classifier derived from attributes and applied on features. Whereas, $X_s W$ (i.e., $[x_1^s; \ldots; x_{n_s}^s]W$) can be treated as the class-classifier derived from features and applied on attributes. In addition, we adopt square loss for $\mathcal{L}$ (i.e., $\mathcal{L}(A, B) = \|A-B\|_F^2$) and ridge regularization for $\mathcal{R}$ (i.e., $\mathcal{R}(A) = \|A\|_F^2$).

For the target domain feature selection part (Eq. 5), we also choose square loss for O, and select a linear model for g, i.e., $g(x_j^t) = x_j^t V$, where $V \in \mathbb{R}^{d,c_t}$ is the generation function. Furthermore, to perform feature selection, we adopt the classical sparse regularization $\ell_{2,1}$-norm [52] for Ω (i.e., $\Omega(V) = \|V\|_{2,1} = \sum_{i=1}^{d}(\sum_{j=1}^{c_t} V^2(i,j))^{\frac{1}{2}} = \sum_{i=1}^{d} \|V(i,:)\|_2$). Theoretically, the $\ell_{2,1}$-norm constraint applied in regression is equivalent to applying Laplace prior [64] on V, which tends to force many rows in V to be 0. More specifically, row $V(i,:)$ shrinks towards zero if the i-th feature is less correlated to the pseudo labels. Therefore, we can filter out the features corresponding to zero rows of V when performing feature selection.

The above two customized parts lead to the specific objective function of the proposed approach:

$$\begin{aligned} \min_{W,Y_t,V} \ & \mathcal{J}_{\text{CKTFS}} = \|Y_s - X_s W S_s\|_F^2 + \|Y_t - X_t W S_t\|_F^2 + \alpha Tr(Y_t' L Y_t) \\ & + \beta \|Y_t - X_t V\|_F^2 + \lambda(\|WS\|_F^2 + \|V\|_{2,1}) \\ \text{s.t.} \ & \|Y_t(j,:)\|_0 = Y_t(j,:)\mathbf{1}_{c_t} = 1, \ j = 1, \ldots, n_t \end{aligned} \tag{7}$$

where $S = [S_s, S_t]$ is the attribute matrix of all classes.

4 OPTIMIZATION SOLUTION

In this section, we propose an effective solution for the optimization issue (Eq. 7) in our CKTFS framework. In addition, we theoretically analyse the time complexity of the proposed optimization method.

4.1 Optimization Algorithm

The objective function in Eq. 7 is a standard quadratic programming problem with 0/1 constraints, which might be hard to solve by the existing convex optimization tools [37] [3]. In this subsection, we give an effective optimization algorithm. Specifically, like [75], we utilize an iterative approach to optimize Eq. 7 until convergence. In each iteration, we update one variable while fixing the other two variables. In addition, recent studies [23] show that a good initiation leads to improved outcomes as well as accelerated convergence. In this paper, we use the ZSL classifier shown in Eq. 2 to generate the initial pseudo label matrix Y_t.

Fix Y_t, V and refine W First, we re-denote the notations: $X = [X_s; X_t]$, $Y = \begin{bmatrix} Y_s & 0_{n_s \times c_t} \\ 0_{n_t \times c_s} & Y_t \end{bmatrix}$, $S = [S_s, S_t]$.

To simplify the problem, we *approximate* the objective function in Eq. 7 w.r.t. W as follows:

$$\min_{W} \mathcal{J}_W = \|Y - XWS\|_F^2 + \lambda \|WS\|_F^2 \tag{8}$$

The derivative of $\mathcal{J}_W$ w.r.t. W is:

$$\frac{\partial \mathcal{J}_W}{\partial W} = 2(-X'YS' + X'XWSS' + \lambda WSS') \tag{9}$$

By setting the above derivative to 0, we obtain the solution for W:

$$W = (X'X + \lambda I)^{-1}(X'YS')(SS')^{-1} \tag{10}$$

Fix W, V and refine Y_t Optimizing Eq. 7 w.r.t. Y_t yields the equation:

$$\begin{aligned} \min_{Y_t} \ & \mathcal{J}_{Y_t} = \|Y_t - X_t WS_t\|_F^2 + \alpha Tr(Y_t' L Y_t) + \beta \|Y_t - X_t V\|_F^2 \\ \text{s.t.} \ & \|Y_t(j,:)\|_0 = Y_t(j,:)\mathbf{1}_{c_t} = 1, \ j = 1, \ldots, n_t \end{aligned} \tag{11}$$

Although Eq. 11 is a convex function, its optimization remains difficult due to the indicator-based integer constraints over Y_t [41]. Therefore, we first get its continuous optimal solution, and then get an approximate solution. Specifically, the derivative of $\mathcal{J}_{Y_t}$ w.r.t. Y_t is:

$$\frac{\partial \mathcal{J}}{\partial Y_t} = 2Y_t - 2X_t WS_t + 2\alpha LY_t + 2\beta Y_t - 2\beta X_t V \tag{12}$$

By setting this derivation to zero, we get the optimal solution for updating Y_t:

$$Y_t = (I + \alpha L + \beta I)^{-1}(X_t WS_t + \beta X_t V) \tag{13}$$

Then, to satisfy the constraint in Eq. 11, we further round Y_t back into an indicator matrix for an *approximate* solution:

$$Y_t(i,j) = \begin{cases} 1, & \text{if } Y_t(i,j) \text{ is the highest value in } Y_t(i,:) \\ 0, & \text{otherwise} \end{cases} \tag{14}$$

Fix W, Y_t and refine V Since the cost term $\|V\|_{2,1}$ in Eq. 7 is not smooth, we follow [52] to relax this term by $Tr(V^T DV)$, where D is a diagonal matrix with its diagonal element $D_{ii} = \frac{1}{2\sqrt{V_{i:}^T V_{i:} + \epsilon}}$ and ϵ is a small positive constant. Then, optimizing Eq. 7 w.r.t. V yields the equation:

$$\min_{V} \mathcal{J}_V = \beta \|Y_t - X_t V\|_F^2 + \lambda Tr(V^T DV) \tag{15}$$

The derivation of $\mathcal{J}_V$ w.r.t. V is:

$$\frac{\partial \mathcal{J}_V}{\partial V} = -2\beta X_t' Y_t + 2\beta X_t' X_t V + 2\lambda DV \tag{16}$$

By setting the derivation to zero, we get the solution for V:

$$V = (\beta X_t' X_t + \lambda D)^{-1}(\beta X_t' Y_t) \tag{17}$$

We can iterate the above three refinements until convergence. After that, we can perform feature selection using the resulted V, i.e., we rank each feature according to the value of $\|V_{i:}\|_2$ in a descending order and return the top ranked features. The complete details of this optimization approach are presented in Alg. 1.

Algorithm 1: Cross-class Knowledge Transfer Feature Selection (CKTFS)

Input: Source instances X_s, label indicators Y_s and source attributes S_s;
Target instances X_t and target attributes S_t;
Parameters α, β and λ;
Convergence criterion ε;

Output: Top ranked features of target data;

1 Initialize Y_t by the ZSL classifier shown in Eq. 2 ;
2 **repeat**
3 Update W by Eq. 10 ;
4 Update Y_t by Eq. 13 and Eq. 14 ;
5 Update V by Eq. 17 ;
6 Update the diagonal matrix D where the i-th diagonal element is $\frac{1}{2\sqrt{V_{i:}^T V_{i:}+\epsilon}}$;
7 **until** $\frac{|\mathcal{J}_{new}-\mathcal{J}_{old}|}{\mathcal{J}_{old}} > \varepsilon$;
8 **return** Rank features according to $\|V_{i:}\|_2$ in a descending order and return the top ranked features.

4.2 Time Complexity

Suppose the total instance number is $N = n_s + n_t$, the total class number is $c = c_s + c_t$, and a is the attribute number. Lines 3, 4 and 5 in Alg. 1 list three main operations of the proposed CKTFS method, and the time complexity of each operation could be computed as:

- Line 3: Updating W involves the inversion of two matrices and some matrix multiplications, and the total time complexity is $O(d^3+d^2N+dcN+a^3)$. This might be inefficient as the practical applications usually contain high-dimensional data (i.e, d may be quite large). According to [52], this operation can be efficiently obtained through solving the linear equation:
$$(X'X + \lambda I)W = (X'YS')(SS')^{-1}$$
whose time complexity is $O(ad^2 + d^2N + dcN + a^3)$.
- Line 4: Updating Y_t requires the inversion of an n_t-by-n_t matrix together with some matrix multiplications. The total time complexity is $O(n_t^3 + dc_t n_t)$, which is very time consuming. Similarly, this can be efficiently obtained through solving the following linear equation:
$$(I + \alpha L + \beta I)Y_t = (X_t W S_t + \beta X_t V)$$
whose time complexity is $O(c_t n_t^2 + dc_t n_t)$.
- Line 5: Updating V involves the inversion of a d-by-d matrix together with some matrix multiplications, whose total time complexity is $O(d^3 + d^2 n_t + dc_t n_t)$. Similarly, we can also turn this matrix inversion operation into solving the following linear equation:
$$(\beta X_t' X_t + \lambda D)V = (\beta X_t' Y_t)$$
whose time complexity is $O(c_t d^2 + d^2 n_t + dc_t n_t)$.

Generally speaking, the attribute number and class number are much less than the feature number and instance number, i.e., $a, c \ll d, N \ll n_t^2$. Therefore, if we all adopt matrix inversion for updating every variable, the total time complexity of each iteration in Alg. 1 is $O(d^3 + d^2N + n_t^3)$. On the other hand, if we achieve these inversions by solving different linear equations, the time complexity of each iteration would become $O(ad^2 + d^2N + c_t n_t^2)$. Besides, in the following experimental section, we can see our algorithm converges fast, usually in less than 15 iterations.

5 THE EXTENSIONS OF OUR FRAMEWORK

In this section, we discuss two possible extensions of our framework (i.e., Eq. 6). Specifically, we discuss 1) how to involve non-linear models, and 2) how to deal with the case where source and target classes are partly overlapping.

5.1 Involving Non-linear Model

Although we can directly customize our framework (i.e., Eq. 6) with non-linear models, this may dramatically increase the optimization difficulty of our method. Therefore, we consider this extension in two subsequent steps. The first step is the cross-class knowledge transfer step (Section 3.2.1), in which we can first learn a non-linear cross-class knowledge transfer model (in Eq. 2). For example, we can follow [31] to train a simple Multilayer Perception model (i.e., using the fixed input features, and embedding class-attribute descriptions in the softmax layer) on the source domain for cross-class knowledge transfer. After that, with the learned non-linear model, we could obtain the predicted target instance labels $\hat{Y}_t$[3].

In the second step, we jointly consider the transferred knowledge (i.e., $\hat{Y}_t$) and target domain feature selection. Specifically, in the feature selection part (formulated in Eq. 4 and Eq. 5), we hope that the pseudo labels (i.e., Y_t) should not change too much compared to the transferred knowledge (i.e., $\hat{Y}_t$). This yields the following optimization problem:

$$\begin{aligned} \min_{Y_t, V} \ & \mathcal{J} = \left\| Y_t - \hat{Y}_t \right\|_F^2 + \alpha Tr(Y_t' L Y_t) + \beta \left\| Y_t - X_t V \right\|_F^2 + \lambda \left\| V \right\|_{2,1} \\ \text{s.t.} \ & \left\| Y_t(j,:) \right\|_0 = Y_t(j,:)\mathbf{1}_{c_t} = 1, \ j = 1, \ldots, n_t \end{aligned} \tag{18}$$

We call this extended method as CKTFS_{nl}, as it involves non-linear models.

Optimization The optimization approach for the problem in Eq. 18 is almost the same as Alg. 1. The only difference is the way of updating Y_t as given W and V. Specifically, by setting the derivation of the objective function Eq. 18 w.r.t. Y_t to zero, we get the optimal solution for updating Y_t:

$$Y_t = (I + \alpha L + \beta I)^{-1}(\hat{Y}_t + \beta X_t V) \tag{19}$$

Then, to satisfy the 0/1 constraints in Eq. 18, we further round Y_t back into an indicator matrix as described in Eq. 13.

5.2 Dealing with the overlap of source and target domains

In this subsection, we consider the case where source and target classes are partly overlapping, i.e., $C_s \cap C_t \neq \varnothing$. In other words, the source domain contains some target classes' labeled instances. For convenience, we can first move these (labeled) target class instances from the source domain to target domain. Now, the target data set X_t can be divided into two subsets. The first part is a set of n_{tl} labeled instances $X_{tl} = \{x_1^t, \ldots, x_{n_{tl}}^t\} \in \mathbb{R}^{n_{tl}, d}$ which is associated with class labels $\widetilde{Y}_{tl} \in \{0,1\}^{n_{tl}, c_t}$. The second part is a set of n_{tu}=n_t−n_{tl} unlabeled instances $X_{tu} = \{x_{l+1}^t, \ldots, x_{n_t}^t\} \in \mathbb{R}^{n_{tu}, d}$ whose labels $\widetilde{Y}_{tu} \in \{0,1\}^{n_{tu}, c_t}$ is unknown, i.e., $\widetilde{Y}_{tu} = 0^{n_{tu}, c_t}$. Similarly, the predicted target data label matrix Y_t can also be rearranged as $[Y_{tl}; Y_{tu}]$. In particular, $Y_{tl} \in \{0,1\}^{n_{tl}, c_t}$ is the predicted label of n_{tl} labeled instances (i.e., X_{tl}) and $Y_{tu} \in \{0,1\}^{n_{tu}, c_t}$ is the predicted label of n_{tu} unlabeled instances (i.e., X_{tu}).

As in this case some target instances are labeled, we should ensure the predicted target data labels (i.e., Y_{tl}) to be consistent with this knowledge. Therefore, we add a new constraint (i.e.,

[3]To avoid ambiguity, we use $\hat{Y}_t$ to denote the predicted target labels in this first step, and continue to use Y_t to denote the pseudo labels used in the feature selection part (in Section 3.2.2).

$Y_{tl} = \widetilde{Y}_{tl}$) into our original objective function (i.e., Eq 7). This yields the following optimization objective function:

$$\begin{aligned} \min_{W,Y_t,V} \quad & \mathcal{J}_{\text{CKTFS}} \\ \text{s.t.} \quad & \|Y_t(j,:)\|_0 = Y_t(j,:)\mathbf{1}_{c_t} = 1,\ j = 1,\ldots,n_t \\ & Y_{tl} = \widetilde{Y}_{tl} \end{aligned} \tag{20}$$

We call this extended method as CKTFS_o, as it considers the overlap case of source and target domains.

Optimization The optimization approach for the problem in Eq. 20 is almost the same as Alg. 1. The main difference is that: after updating Y_t as Alg. 1 does, we further force $Y_{tl} = \widetilde{Y}_{tl}$ to satisfy the constraint of this problem. Besides, for the same reason, we also force $Y_{tl} = \widetilde{Y}_{tl}$ when initializing Y_t at the beginning of Alg. 1.

6 RELATIONS TO OTHER APPROACHES

In this section, we discuss the relationship between some existing feature selection methods and our methods (including CKTFS, CKTFS_nl and CKTFS_o). Note: in the following, we first remove the cross-class knowledge transfer part of our method (formulated in Eq. 7) and then compare ours with others. This is because our method is the first method which transfers the cross-class knowledge to guide feature selection.

6.1 The relation of CKTFS to other approaches

After removing the cross-class knowledge transfer part, the objective function of CKTFS (formulated in Eq. 7) becomes:

$$\begin{aligned} \min_{Y_t,V} \quad & \alpha Tr(Y_t'LY_t) + \beta \|Y_t - X_tV\|_F^2 + \lambda \|V\|_{2,1} \\ \text{s.t.} \quad & \|Y_t(j,:)\|_0 = Y_t(j,:)\mathbf{1}_{c_t} = 1,\ j = 1,\ldots,n_t \end{aligned} \tag{21}$$

Relation to JELSR The well-known unsupervised feature selection method JELSR [27, 28] joins embedding learning with sparse regression for feature selection, which can be formulated as the following problem:

$$\min_{Y_t'Y_t = I_{c_t,c_t},V} Tr(Y_t'LY_t) + \beta \|Y_t - X_tV\|_F^2 + \lambda \|V\|_{2,1} \tag{22}$$

where I_{c_t,c_t} is a $c_t \times c_t$ identity matrix.

Comparing the formulation in Eq. 22 with that in Eq. 21, we can see the main difference lies in two points. The first point is the constraint imposed on Y_t, i.e., the 0/1 constraint in our method and orthogonality constraint in JELSR. In fact, this orthogonality constraint is well-known as a relaxation of the 0/1 constraint [72]. The second point is the way of building the Laplacian matrix (i.e., L): JELSR employs the locally linear approximation and our method employs binary k-nearest-neighbor strategy. Consequently, JELSR in Eq. 22 can be regarded as a special case of CKTFS, if we remove the cross-class knowledge transfer part of our method, employ the locally linear approximation strategy in Eq. 4, and relax the 0/1 constraint in Eq. 3 as $Y_t'Y_t = I_{c_t,c_t}$.

Relation to MCFS MCFS [5] is also a well-known unsupervised feature selection approach. It first adopts the spectral clustering to characterize manifold structure, and then perform feature selection via ℓ_1-norm regularized regression learning. Specifically, MCFS solves the following optimization problem in a two stage way:

$$\begin{aligned} & \min_{Y_t'DY_t = I_{c_t,c_t}} Tr(Y_t'LY_t) \\ & \min_{V} \|Y_t - X_tV\|_F^2 + \lambda \|V\|_{1,1} \end{aligned} \tag{23}$$

Comparing the formulation in Eq. 23 with that in Eq. 21, we can see the main difference lies in three points. Firstly, MCFS has a D-orthogonal constraint for Y_t (i.e., $Y_t'DY_t = I_{c_t,c_t}$), which can be seen as a relaxation and re-normalization of the 0/1 constrain in Eq. 21. Secondly, MCFS adopts the ℓ_1-norm regularization for regression learning (i.e., feature selection learning). Thirdly, MCFS separates the manifold learning and regression learning. Therefore, MCFS in Eq. 23 can be regarded as a special case of CKTFS, if we remove the cross-class knowledge transfer part, and replace the 0/1 constraint and $\ell_{2,1}$-norm regularization in Eq. 7 with the D-orthogonal constraint and ℓ_1-norm regularization respectively, and finally separate the manifold learning and sparse learning.

Relation to Spectral Regression Spectral Regression (SR) [4] is a famous dimensionality reduction approach. It first adopts the spectral clustering (i.e., the idea of graph laplacian) to characterize manifold structure, and then learns a linear dimensionality reduction function via regression. Theoretically, SR solves the following optimization problem in a two stage way:

$$\begin{aligned} &\min_{Y_t'DY_t=I_{c_t,c_t}} \quad Tr(Y_t'LY_t) \\ &\min_{V} \quad \|Y_t - X_tV\|_F^2 + \lambda \|V\|_2^2 \end{aligned} \tag{24}$$

Firstly of all, comparing the formulation in Eq. 24 with that in Eq. 23, we can find the only difference between SR and MCFS is: they adopt different regularization norms. Therefore, and similar to the analysis of MCFS, SR in Eq. 24 can be regarded as a special case of CKTFS, if we remove the cross-class knowledge transfer part, and replace the 0/1 constraint and $\ell_{2,1}$-norm regularization in Eq. 7 with the D-orthogonal constraint and $\ell_{2,2}$-norm regularization respectively, and finally separate the manifold learning and the $\ell_{2,2}$-norm regularized regression learning.

6.2 The relation of CKTFS$_{\text{nl}}$ to other approaches

As mentioned in Section 5.1, comparing to the original method, CKTFS$_{\text{nl}}$ performs the cross-class knowledge transfer task and target domain feature selection task in sequence. Therefore, we omit the discussion about CKTFS$_{\text{nl}}$ whose analysis would be similar to that of CKTFS.

6.3 The relation of CKTFS$_{\text{o}}$ to other approaches

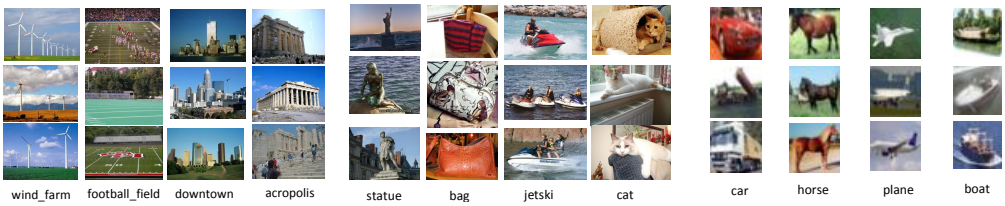
After removing the cross-class knowledge transfer part, the objective function of CKTFS$_{\text{o}}$ (formulated in Eq. 20) becomes:

$$\begin{aligned} \min_{Y_t,V} \ & \alpha Tr(Y_t'LY_t) + \beta \|Y_t - X_tV\|_F^2 + \lambda \|V\|_{2,1} \\ \text{s.t.} \ & \|Y_t(j,:)\|_0 = Y_t(j,:)\mathbf{1}_{c_t} = 1, \ j = 1, \ldots, n_t \\ & Y_{tl} = \widetilde{Y}_{tl} \end{aligned} \tag{25}$$

Relation to SFSS SFSS [50] is a well-known semi-supervised feature selection method. Specifically, it joins feature selection, manifold regularization and transductive classification. Mathematically, this method can be regarded as solving the following problem:

$$\min_{Y_t,V} \ Tr(Y_t'LY_t) + Tr((Y_t - \widetilde{Y}_t)'U(Y_t - \widetilde{Y}_t)) + \beta \|Y_t - X_tV\|_F^2 + \lambda \|V\|_{2,1} \tag{26}$$

where $\widetilde{Y}_t = [\widetilde{Y}_{tl}; \widetilde{Y}_{tu}]$ is the given label information of target data, and U is a diagonal matrix whose diagonal element U_{ii} is a very large value (like 10^{10}) if target instance x_i^t is labeled and $U_{ii} = 1$ otherwise. Intuitively, matrix U is defined to make the predicted labels Y_t consistent with the given labels $\widetilde{Y}_t$.



(a) SUN (b) aPY (c) CIFAR10

Fig. 3. Some samples from the tested image datasets. (a) SUN. (b) aPY. (c) CIFAR10. (AwA does not provide the image set in JPEG format for copyright reasons.)

Theoretically, when we set $U_{ii} = \infty$ for every labeled target instance x_i^t, optimizing Eq. 26 is equivalent to optimizing the following optimization problem:

$$\min_{Y_t, V, Y_{tl}=\widetilde{Y}_{tl}} Tr(Y_t' L Y_t) + \beta \left\| Y_t - X_t V \right\|_F^2 + \lambda \left\| V \right\|_{2,1} \tag{27}$$

Comparing Eq. 27 and Eq. 25, the only difference is that Eq. 25 adds extra 0/1 constraint on the unlabeled target data's predicted labels (i.e., Y_{tu}). Therefore, SFSS in Eq. 26 can be regarded as a special case of CKTFS$_o$, if we remove the cross-class knowledge transfer part and the 0/1 constraint in CKTFS$_o$.

Relation to LASSO LASSO [70] is the classical supervised feature selection method. It learns an ℓ_1-norm regularized linear least squares regression, which can be formulated as:

$$\min_{V} \left\| \widetilde{Y}_{tl} - X_{tl} V \right\|_F^2 + \lambda \left\| V \right\|_{1,1} \tag{28}$$

We can clearly find that LASSO in Eq. 28 can be regarded as a special case of CKTFS$_o$, if we remove the cross-class knowledge transfer part and manifold learning part, and only consider the labeled target data, and replace $\ell_{2,1}$-norm regularization with ℓ_1-norm in CKTFS$_o$.

7 EXPERIMENTS

In this section, we conduct extensive experiments to evaluate the performance of the proposed CKTFS method. Firstly, we introduce four real-world datasets. Then, since existing methods cannot address the cross-class setting, we compare the proposed method with both unsupervised and supervised feature selection methods for an extensive comparison. After that, we evaluate the performance of our method w.r.t. source domain size. Lastly, we study the convergence of the proposed optimization algorithm (Alg. 1) and the effect of parameters on performance.

7.1 Experimental Setup

Datasets We test the proposed CKTFS method on four benchmark datasets with attributes. SUN scene attributes database (**SUN**) [57] [4] is the first dataset, which contains 14,340 images from 717 different scenes like "beach" and "airport". In this dataset, each class has 20 images, and each image is annotated by a 102-dimensional binary attribute vector manually. We obtain the attribute representation of each class by averaging attribute vectors of the images belonging to that class. We follow the source/target split in [30].

[4] https://cs.brown.edu/ gen/sunattributes.html

Table 2. The statistics of datasets

	SUN	aPY	AwA	CIFAR10
# source classes	707	20	40	5
# source images	14,140	12,695	24,295	30,000
# target classes	10	12	10	5
# target images	200	2,644	6,180	30,000
# attributes	102	64	85	50
# features	4,096	4,096	4,096	4,096

The second dataset is aPascal/aYahoo objects dataset (**aPY**) [20][5] which has two parts. The aPascal part contains 20 types of objects from the PASCAL VOC2008 dataset, and the aYahoo part contains 12 different categories collected using Yahoo image search engine. We follow the standard source/target split setting where the aPascal part and aYahoo part serves as the source and target domain, respectively. In this dataset, each image is annotated by a 64-dimensional binary attribute vector. We average the attribute representations of images in the same category to get the class-attribute representation.

The third dataset is Animal with Attributes (**AwA**) [35][6] which contains 30,475 images from 50 animal classes such as "zebra" and "mouse". In this dataset, each class is described by a 85-dimensional binary attribute vector including various attributes such as "is black" and "eats fish". We adopt its standard source/target split, i.e., 40 classes with 24,295 images are adopted as the source part and 10 classes with 6,180 images are adopted as the target part.

The fourth dataset is **CIFAR10** [33][7] which consists of 10 classes of objects with 6,000 images in each class. Since this dataset does not have a standard source/target split, we simply use the top half classes (i.e., "airplane", "automobile", "bird", "cat" and "deer") as source classes and use the other half as target classes. In addition, since this dataset does not provide any manually annotated attributes, we adopt the 50-dimensional class-label embedding vector provided by [29] as attributes[8]. Concretely, these embedding vectors are automatically generated from a Wikipedia corpus [65] with a total of about 2 million articles and 990 million tokens.

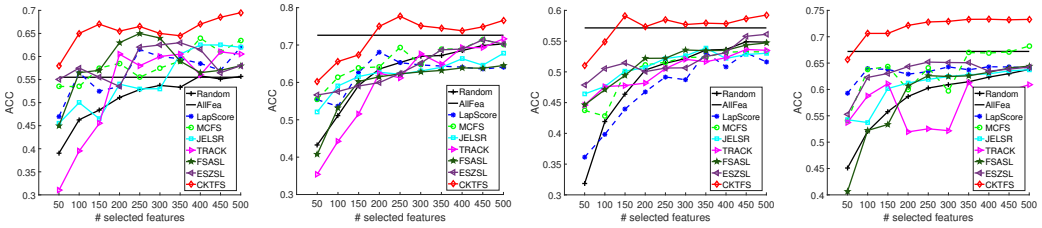
For these four datasets, we all adopt the widely used 4,096-dimensional deep features. Specifically, for SUN, aPY, and CIFAR10, we adopt the deep features provided by [22]. For AwA, we adopt the deep features provided in its project page. The statistics of these four benchmarks are shown in Table 2, and some samples can be found in Fig. 3.

Experimental Setting Since determining the optimal number of selected features is still an open problem [69], we follow [48] to select the number of features ranging from 50 to 500 with incremental step 50. The quality of the selected features by various methods is evaluated in the tasks of both clustering and classification. Specifically, we directly apply the compared methods on the target data to compare with ours, since existing methods cannot address the cross-class setting.

7.2 Comparison to Unsupervised Methods

In this subsection, we compare the proposed method with state-of-the-art unsupervised feature selection methods. Following a standard way to assess unsupervised feature selection [25], we evaluate the quality of the selected features in terms of clustering performance. Specifically, each

[5]http://vision.cs.uiuc.edu/attributes/
[6]http://www.attributes.kyb.tuebingen.mpg.de/
[7]https://www.cs.toronto.edu/ kriz/cifar.html
[8]https://github.com/ninjin/huang_et_al_2012



(a) SUN (b) aPY (c) AwA (d) CIFAR10

Fig. 4. Clustering results (ACC).

compared method is first applied to select features in the target domain. Then the classical K-means clustering[9] is performed on target data represented by the selected features. Since K-means is sensitive to initialization, we repeat the clustering 20 times with random initializations and record the average performance. Finally, as in [15], the clustering result is evaluated by two widely used clustering quality evaluation metrics, Accuracy (ACC) and Normalized Mutual Information (NMI)[10] [17].

Baselines In this study, we compare the proposed method with the following baselines:

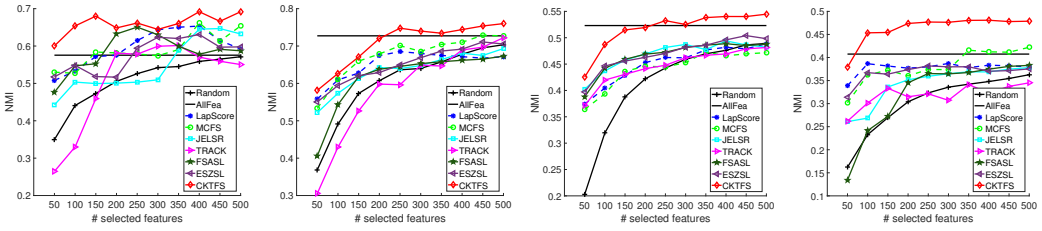
(1) Random which selects features randomly.
(2) AllFea which selects all original features.
(3) LapScore [25] which evaluates features according to their ability of preserving the local manifold structure.
(4) MCFS [5] which selects features by using spectral regression with ℓ_1-norm regularization.
(5) JELSR [28] [27] which joins embedding learning with sparse regression for feature selection.
(6) TRACK [73] which selects features via a unified trace ratio formulation and K-means clustering.
(7) ESZSL [60] which is a ZSL method. For feature selection, we first adopt ESZSL to predict target instance labels, and then feed these labels to the classical supervised feature selection method LASSO [70].
(8) FSASL [15] which performs structure learning and feature selection simultaneously.

Parameter Settings Following [5], for LapScore, MCFS, JELSR, and the proposed CKTFS method, we fix the neighborhood size to be 5. As suggested in [73], the reduced dimension m in TRACK is set as: $m{=}c{-}1$ if $d{\leq}n$, and $m{=}c{-}1{+}d{-}n$ if $d{>}n$, where c is the target class number and d is the data dimensionality. Similar to [15, 27], to fairly compare all methods, we tune their parameters (if any) by a "grid-search" strategy from $\{10^{-2}, 10^{-1}, 10^{0}, 10^{1}, 10^{2}\}$ and report the best results. For the ZSL method ESZSL, we treat the source domain as seen and target domain as unseen, so as to following the general sitting of ZSL studies [1]. In our method, our parameters are chosen (from the above "grid") via cross-validation on the source domain.

Experimental Results Figure 4 and Figure 5 show the clustering performance in terms of ACC and NMI, respectively. There are several important observations as follows.

The first observation is that: compared with AllFea (i.e., using all features), feature selection is necessary and effective by removing the noise and redundancy. In particular, it not only reduces feature number but also improves the performance of K-means clustering. For example, in the

[9]We adopt the source code from http://www.cad.zju.edu.cn/home/dengcai/Data /code/litekmeans.m
[10]We adopt the source code from www.cad.zju.edu.cn/home/dengcai/ Data/Clustering.html



(a) SUN (b) aPY (c) AwA (d) CIFAR10

Fig. 5. Clustering results (NMI).

SUN dataset, when using only 200 (of the total 4099) features, almost all feature selection methods outperform AllFea. Besides, with only 200 features, the proposed CKTFS method could achieve competitive or even better performance than AllFea in all four datasets. This further shows the superiority of our method. On the other hand, we also find that the clustering performance does not always increase with more selected features. This indicates that the original features contain some redundancy and noise, which could be harmful for data processing applications.

The second observation is that, on the one hand, ESZSL clearly outperforms Random; while on the other hand, ESZSL still performs much worse than our method CKTFS. First of all, the comparison between ESZSL and Random verifies the usefulness of attribute-based cross-class transferred knowledge for feature selection. We can further explain this by looking at the target domain classification accuracy of ESZSL. As shown in Table. 3, ESZSL achieves much higher accuracy than Random-Guess (i.e., classifying randomly) on the target domain. On the other hand, the comparison between ESZSL and CKTFS indicates that ESZSL could be further improved by considering the manifold structure of target domain. Moreover, we also find some unsupervised methods (like MCFS and JELSR) achieve competitive or even better performance than ESZSL. All these observations confirm the advantage of our basic idea, i.e., considering both cross-class knowledge transfer and target data manifold learning.

The third observation is that the proposed CKTFS method significantly better than those unsupervised methods, in terms of both ACC and NMI, on all datasets with different numbers of selected features. For example, when selecting 100 features in all four datasets, CKTFS outperforms the best baseline approaches JELSR or MCFS by 20~50% relatively in terms of both ACC and NMI. This is a remarkable result for unsupervised feature selection. The major reason is that CKTFS benefits from the success of attribute based cross-class knowledge transfer. Specifically, the introduced pseudo labels in CKTFS encode the cross-class transferred knowledge. As a counterexample, the compared method JELSR, which also uses pseudo labels but fail to capture this knowledge, performs significantly worse than ours.

The fourth observation is that CKTFS also outperforms baseline methods even when attributes are automatically generated. In particular, Figs. 4(d) and 5(d) show the experimental results on the CIFAR10 dataset whose attributes are automatically generated from a Wikipedia corpus. On average, CKTFS outperforms the best baseline methods TRACK or MCFS around 15% relatively in terms of both ACC and NMI. These results indicate that our method has a wide application range.

Finally, we report the statistical significance of these comparison results shown in Fig. 4 and Fig. 5. Following [27], we compare our method with other baselines by Student's t-test. To avoid redundancy, we only report the averaged comparison results with two baselines: AllFea and the best compared feature selection method. As shown in Table 4, from the statistical view, we can see that CKTFS achieves significantly better results comparing to the other methods. Moreover, in

Table 3. ZSL classification accuracy on target domain.

	SUN	aPY	AwA	CIFAR10
Random-Guess	0.0950	0.0859	0.1073	0.1993
ESZSL	**0.6900**	**0.1700**	**0.3683**	**0.4101**

Table 4. The t-Test results between our method CKTFS and other methods for the averaged clustering results in Figs. 4 and 5. "W" means CKTFS performs better. "F" means other method performs better. "B" means that CKTFS and other method cannot outperform each other. The value in the bracket is the associate p-Value (the smaller p-Value means the higher assurance of the conclusion). Statistical significance of t-Test is 5%, and we do not report the p-Value when the mark is "B".

	SUN		aPY		AwA		CIFAR10	
Top2 Baselines	AllFea	FSASL	AllFea	MCFS	AllFea	FSASL	AllFea	MCFS
ACC	W(.00)	W(.00)	F(.01)	W(.00)	B(-)	W(.00)	W(.00)	W(.00)
NMI	W(.00)	W(.00)	F(.00)	W(.00)	F(.01)	W(.00)	W(.00)	W(.00)

some cases, our method even achieves significant better results than AllFea. These results reaffirm the effectiveness of our method.

7.3 Comparison to Supervised Methods

In this subsection, we compare the proposed method with state-of-the-art supervised feature selection methods.

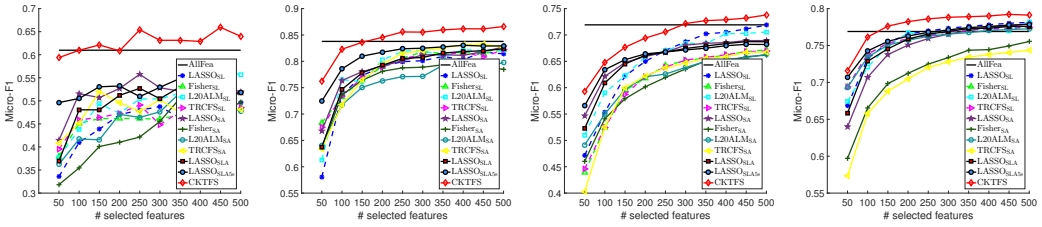
Baselines We compare the proposed method with the following baselines:

(1) AllFea which selects all original features.
(2) LASSO [70] which is the classical feature selection method based on the ℓ_1-norm regularization.
(3) Fisher [16] which selects features with large between-class distance and small within-class distance.
(4) L20ALM [6] which evaluates features under an ℓ_{21}-norm loss function with ℓ_{20}-norm constraint.
(5) TRCFS [48] which is a recent semi-supervised feature selection algorithm based on noise in-sensitive trace ratio criterion.

First and foremost, although we assume target data contains no supervision information, the class labels and attribute information in the source domain can be treated as supervision information. In addition, to make a comprehensive comparison, we even provide a few labeled target instances to test the performance of these baselines. More specifically, we test the following settings of a compared supervised method $\mathcal{M}$:

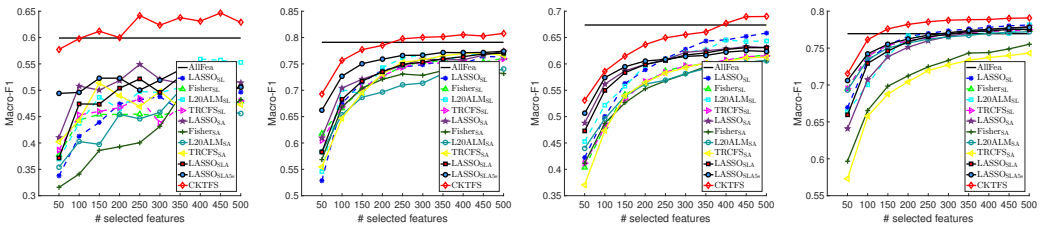
- $\mathcal{M}_{\mathrm{SL}}$: adopting the source class labels (i.e., Y_s) as supervision information.
- $\mathcal{M}_{\mathrm{SA}}$: adopting the source class attributes (i.e., S_s) as supervision information.
- $\mathcal{M}_{\mathrm{SLA}}$: adopting both source class labels and attributes as supervision information.
- $\mathcal{M}_{\mathrm{SLA5s}}$: adopting source class labels, source class attributes and 5 labeled instances of each target class as supervision information.

Following a standard way to assess supervised feature selection [48], we use classification performance to evaluate the quality of the selected features. For each compared supervised method $\mathcal{M}$, we first obtain some important features in a specific setting (like $\mathcal{M}_{\mathrm{SL}}$). Then, in the target domain, we test the quality of the selected features by the popular 1-nearest neighbor classifier



(a) SUN (b) aPY (c) AwA (d) CIFAR10

Fig. 6. Classification results (Micro-F1).



(a) SUN (b) aPY (c) AwA (d) CIFAR10

Fig. 7. Classification results (Macro-F1).

as [6]. Specifically, as [6], we randomly select 20% of target data for training this classifier and use the remaining part for testing. The reason of using small portion of training data is because it is well-known that when the training data size becomes sufficiently large, any feature selection method will perform well. We repeat the classification 20 times, and report the average performance in terms of two commonly used metrics Micro-F1 and Macro-F1 [17].

In addition, for the setting of $\mathcal{M}_{\text{SLA}}$ (i.e., adopting both source class labels and attributes as supervision information), we first obtain the feature importance ranking lists of $\mathcal{M}_{\text{SL}}$ and $\mathcal{M}_{\text{SA}}$ separately. Then, to get the results of $\mathcal{M}_{\text{SLA}}$, we adopt the classical "CombSUM" multiple ranking list combination strategy [39] to combine these two ranking lists. We implement $\mathcal{M}_{\text{SLA}}$ through these two steps, since none of the compared methods could simultaneously utilize two kinds of supervision. Similarly, for the setting of $\mathcal{M}_{\text{SLA5s}}$, we also adopt the "CombSUM" strategy to combine three individual ranking lists (i.e., the results of $\mathcal{M}_{\text{SL}}$, $\mathcal{M}_{\text{SA}}$ and $\mathcal{M}$ with 5 labeled instances of each target class).

Parameter Settings For a fair comparison, we also tune parameters as we do previously. In addition, for L20ALM, the suggested parameter setting ($\mu = 0.01$ and $\rho = 1.02$) is also tested. For all compared methods, we report the best results we could obtain.

Experimental Results Figure 6 and Figure 7 show the classification accuracy in terms of Micro-F1 and Macro-F1, respectively. The results are similar to those of the clustering task, i.e., CKTFS almost always outperforms other supervised methods, in terms of both Micro-F1 and Macro-F1, on all datasets with different numbers of selected features.

Specifically, firstly, when adopting source class labels as supervision, all compared methods (i.e., $\mathcal{M}_{\text{SL}}$) perform much worse than our method. This indicates that it is inappropriate to directly borrow supervised knowledge form source domain when its classes are different from target ones. The reason might be that in the cross-class setting source class labels only reflect the discrimination

Table 5. The t-Test results between our method CKTFS and other methods for the averaged classification results in Figs. 6 and 7. "W", "F" and "B" denote the same meanings as in Table. 4. We also report the p-Value and the statistical significance of t-Test is also 5%.

	SUN		aPY		AwA		CIFAR10	
Top2 Baselines	AllFea	$\text{LASSO}_{\text{SLA5s}}$	AllFea	$\text{LASSO}_{\text{SLA5s}}$	AllFea	$\text{LASSO}_{\text{SLA5s}}$	AllFea	$\text{LASSO}_{\text{SLA5s}}$
Micro-F1	W(.00)	W(.00)	W(.00)	W(.00)	F(.00)	W(.00)	W(.00)	W(.00)
Macro-F1	W(.00)	W(.00)	B(-)	W(.00)	F(.00)	W(.00)	W(.01)	W(.00)

among source classes, and provide little information about target classes. Therefore, supervised methods may be misled by the source class labels which provide little information about target classes. Intuitively, we can understand this by the fact that old knowledge may often need to be modified to fit the new problem.

Secondly, when adopting source attributes as supervision, all compared methods (i.e., $\mathcal{M}_{\text{SA}}$) still perform much worse than our method. This also shows the superiority of our method. However, an interesting observation is that $\mathcal{M}_{\text{SA}}$ methods mostly perform much better than $\mathcal{M}_{\text{SL}}$ ones. For example, LASSO_{SL} achieves the best performance among all the compared baselines in most cases. This may be partly explained by the success of ZSL which has shown that attributes could serve as a bridge to transfer supervised knowledge across different classes.

Thirdly, our method still outperforms these supervised methods when they adopt two or even three kinds of supervision information (i.e., $\mathcal{M}_{\text{SLA}}$ and $\mathcal{M}_{\text{SLA5s}}$)[11]. For example, although $\text{LASSO}_{\text{SLA5s}}$ even utilizes some target label information, our method still outperforms it by 20∼100% relatively in terms of both Micro-F1 and Macro-F1. These results reaffirm that our method can be an effective way for discriminative feature selection, when few or even no labeled examples are available.

Finally, we also report the statistical significance of these comparison results shown in Fig. 6 and Fig. 7. The comparison setting is the same as that of Section 7.2. As shown in Table 5, from the statistical view, we can see that CKTFS achieves significantly better results comparing to the other feature selection methods. These results reaffirm that our method can select features very effectively.

7.4 More extensions of CKTFS

In this subsection, we test more extensions of our method in the following settings:

- CKTFS_g: adopting Gaussian kernel (i.e., $\mathcal{K}(x_i, x_j) = e^{\frac{-\left\|x_i - x_j\right\|^2}{2\sigma^2}}$) weight for the k-nearest-neighbor graph, so as to capture the structure of target domain (Section 3.2.2). Specifically, following [77], the kernel width σ is empirically set as $2^{1/2.5}\delta$, where δ is the standard deviation of the target data.
- CKTFS_{nl}: adopting non-linear model in the proposed framework (Section 5.1). For simplicity, we adopt the simple Multilayer Perception model described in Section 5.1.
- CKTFS_o: dealing the case where source and target classes are partly overlapping. For simplicity, we randomly select two target classes as partly labeled classes, and adopt 30% the instances of these two classes as labeled data. We repeat this random selection 20 times.

Figure 8 shows the averaged clustering accuracy of all these variants. The best two compared feature selection methods (i.e., MCFS and FSASL) are adopted as baselines. Firstly, we can see that CKTFS_g achieves very similar performance as the original CKTFS method. This indicates that

[11]Due to space limitations, for the settings of $\mathcal{M}_{\text{SLA}}$ and $\mathcal{M}_{\text{SLA5s}}$, we only report the results of LASSO which shows the best performance in the settings of $\mathcal{M}_{\text{SL}}$ and $\mathcal{M}_{\text{SA}}$.

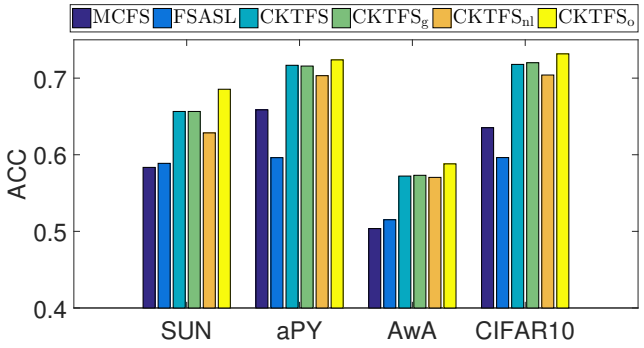


Fig. 8. Averaged clustering performance (ACC) w.r.t different extensions of CKTFS.

Gaussian kernel could also be adopted to capture the manifold structure of target domain, which is consistent with previous studies [27] [83].

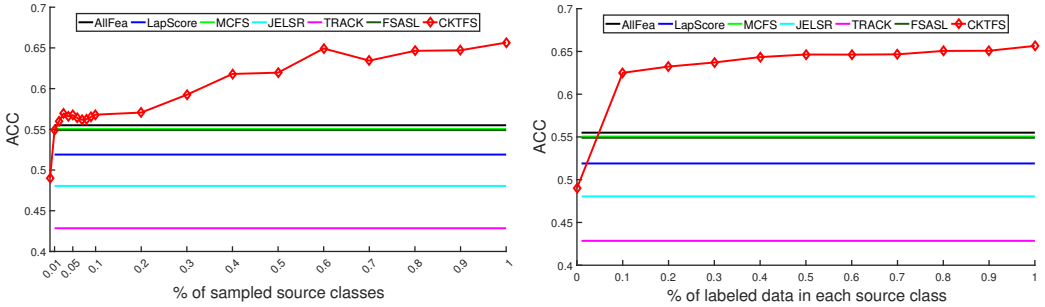
Secondly, on the one hand, we find that CKTFS$_{nl}$ performs worse than the original CKTFS method. The reason might be that CKTFS$_{nl}$ treats the cross-class knowledge transfer and target domain feature selection separately. In contrast, the original CKTFS method treats these two parts in a joint framework for a better solution. This indicates that jointly considering source and target domains is more effective than involving non-linear models in separate steps. On the other hand, we also find that CKTFS$_{nl}$ significantly outperforms the best two compared feature selection methods. This indicates that CKTFS$_{nl}$ still benefits from the success of attribute based cross-class knowledge transfer.

Thirdly, CKTFS$_o$ always achieves the best performance. This shows that CKTFS$_o$ could benefit from the target domain's label information, which also validates that our framework (i.e., Eq. 6) could be extended to deal with the case where source and target classes are partly overlapping.

7.5 Effect of Source Domain Size

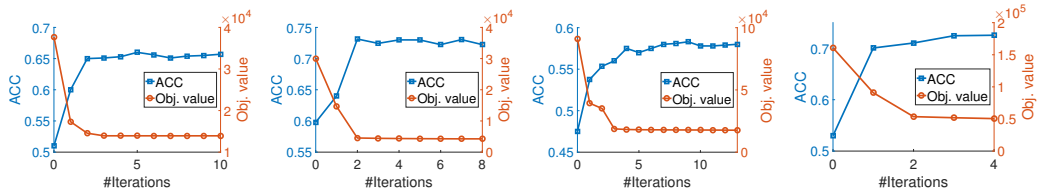
In this part, we evaluate the performance of the proposed CKTFS method w.r.t. source domain size. To make an extensive comparison, we also test CKTFS in an extreme case where the source domain is completely removed. Although, this case is not the focus of this paper, as shown in Eq. 21 in Section 6.1, we can still extend our method to handle this case. Here, we only conduct on the SUN dataset, because we have similar observations in other datasets. On this dataset, since our method reaches the best performance with around 100 features and starts to fluctuate, we change the initial feature setting to a more fine-grained one (i.e., [20,40,...,200]) to show the effect of source domain size more clearly. For simplicity, we experimentally set the parameters in CKTFS as: $\alpha = 0.1$, $\beta = 1$, and $\lambda = 0.01$. For an extensive study, we design two different experiments.

The first experiment is to test the effect of source class number. In particular, we keep the target domain unchanged and randomly choose some source classes for CKTFS. In other words, the number of selected source class is changing while each source class size is unchanged (i.e., 20 images/class). Figure 9(a) shows the aggregated clustering results in terms of ACC. Firstly, we can find that even with only 1% of source classes, our method could still benefit the success of cross-class knowledge transfer. Contrarily, the compared method JELSR, which also utilizes pseudo labels but neglect this knowledge, performs much worse than ours. Secondly, we find that as the source class number grows, the performance of our method grows. We analyze that with more source classes, we can learn a better inter-mediate knowledge, i.e., the relationship between attributes and features. As such, more reliable supervised knowledge would be transferred to the target domain,



(a) Effects of different ratios of source classes. (b) Effects of different labeled ratios in all source classes.

Fig. 9. Effect of Source Domain Size on SUN dataset.



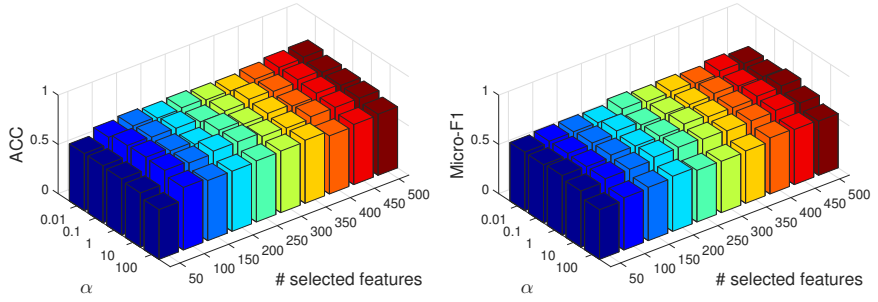
(a) SUN (b) aPY (c) AwA (d) CIFAR10

Fig. 10. Clustering performance and objective function value (in Eq. 7) at different numbers of iterations.

thereby simultaneously improving the quality of selected features. Lastly, when the source class rate reduces to 0%, our method only has the similar performance as JELSR. As theoretically analysed in Section 6.1, this is because after removing the cross-class knowledge transfer part, the formulation of CKTFS reduces to Eq. 21 which can be directly relaxed to JELSR's formulation (i.e., Eq. 22). In summary, the above observations reaffirm the significance of transferring cross-class knowledge for feature selection.

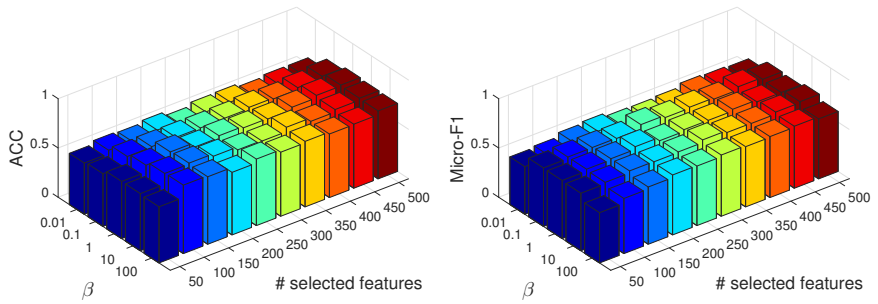
The second experiment is to test the effect of labeled ratio in all source classes. Specifically, we still keep the target domain unchanged and vary the ratio of labeled data in all source classes. In other words, the source class number (i.e., 707) is unchanged while each source class size is changing. Figure 9(b) shows the aggregated clustering results in terms of ACC. It is easy to find and understand that as the ratio of labeled data grows, the performance of CKTFS improves significantly.

Considering the results of these two experiments together, we can find an interesting observation: when the sample ratios in these two experiments are same, the performance of CKTFS in the second experiment is always better. For instance, when the sample ratios are both 0.1, the performance in the second experiment has 8% higher observed improvements than the first one. This indicates that compared to the data size in each class, source class number is more critical for our method. We may explain this as follows. When the source class number is fixed, even if the data size of each class increases, the learned knowledge (about features and attributes) would still be limited to these fixed classes. On the other hand, when more variety of source classes are added, the learned knowledge could reflect a more "real-world" experience, and has better generalizability. This may also explain why CKTFS especially outperforms the other methods in the SUN dataset which has significantly more source classes (i.e., 707), compared to the other three datasets (shown in Sections 7.2 and 7.3).



(a) Clustering (ACC) (b) Classification (Micro-F1)

Fig. 11. The effect of parameter α on the SUN dataset.



(a) Clustering (ACC) (b) Classification (Micro-F1)

Fig. 12. The effect of parameter β on the SUN dataset.

7.6 Convergence Analysis

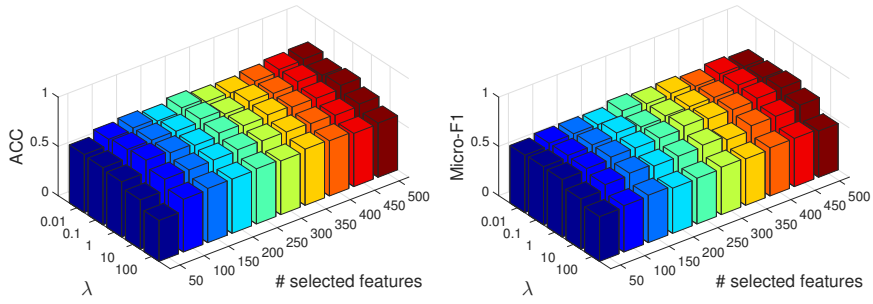
We conduct an empirical experiment to show the effective of the proposed alternating optimization algorithm (Alg. 1). Specifically, we fix the CKTFS's parameters as described in Section 7.5, and empirically set the iteration convergence criterion (i.e., ε) as 10^{-5}. In practice, we can also set a maximum iteration value to control this iteration.

Figure 10 shows the clustering performance and objective function value w.r.t. each iteration in Alg. 1. We can see that CKTFS converges rapidly in all datasets, which is more efficient than some typical existing methods, such as the trace norm [15] and ℓ_{21}-norm based feature selection methods [28]. Meanwhile, the clustering accuracy increases and quickly reaches the highest accuracy in around 5 steps.

7.7 Parameter Analysis

As mentioned in Section 3, the proposed CKTFS method has three parameters. The parameter α controls how strongly the transferred knowledge, i.e., the introduced pseudo labels, preserves the manifold structure of target data. The parameter β measures the fitness of the introduced pseudo labels w.r.t. target features. The third parameter λ, which has been empirically set to 0.01, controls the effect of the regularization term.

In this subsection, we investigate the effect of all these three parameters. Specifically, we vary the value of one parameter while keeping the others fixed at the optimal value. Here, we only



(a) Clustering (ACC) (b) Classification (Micro-F1)

Fig. 13. The effect of parameter λ on the SUN dataset.

present the results on SUN, since we have similar observations in other datasets. The experimental results are shown in Figs. 11, 12 and 13. We have the following observations. i) The performance is consistently improved when more features are selected. This indicates that all the features selected by CKTFS contain additional useful information. ii) CKTFS performs stably and effectively when parameter α lies in $\{10^{-2}, 10^{-1}, 10^{0}\}$. This may be due to the fact that the tested images are all captured from real scenarios and contain lots of noise. In other words, this parameter may be sensitive to the properties of the different datasets. iii) CKTFS reaches high performance, when parameter β gets large values, i.e., $\{10^{0}, 10^{1}, 10^{2}\}$. The reason is quite simple: since our goal is feature selection, we need large β to reflect the relevance between features and pseudo labels. iv) CKTFS is not very sensitive when parameter λ lies in $\{10^{-2}, 10^{-1}, 10^{0}\}$. This suggests us to choose a small regularization parameter, which is in agreement with most related studies, such as [28], [15] and [69].

8 CONCLUSION

This paper investigates the feature selection problem in a cross-class setting where the labeled source classes and unlabeled target classes are related in an intermediate attribute level but different. We propose a novel feature selection framework named CKTFS which transfers the cross-class knowledge from the source domain to guide target domain feature selection. In addition, to further improve the performance, our framework considers the tasks of cross-class knowledge transfer and feature selection jointly. Extensive experiments conducted on several real-world datasets demonstrate the effectiveness of the proposed method. In the future, we plan to extend our method to make it suitable for exploring the label information in the target domain.

REFERENCES

[1] Zeynep Akata, Florent Perronnin, Zaid Harchaoui, and Cordelia Schmid. 2013. Label-embedding for attribute-based classification. In *IEEE Conference on Computer Vision and Pattern Recognition*. 819–826.

[2] Wei Bi, Yuan Shi, and Zhenzhong Lan. 2009. Transferred feature selection. In *IEEE International Conference on Data Mining Workshops*. IEEE, 416–421.

[3] Stephen Boyd and Lieven Vandenberghe. 2004. *Convex Optimization*. Cambridge University Press.

[4] Deng Cai, Xiaofei He, and Jiawei Han. 2007. Spectral regression for efficient regularized subspace learning. In *IEEE International Conference on Computer Vision*. IEEE, 1–8.

[5] Deng Cai, Chiyuan Zhang, and Xiaofei He. 2010. Unsupervised feature selection for multi-cluster data. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 333–342.

[6] Xiao Cai, Feiping Nie, and Heng Huang. 2013. Exact Top-k Feature Selection via l2, 0-Norm Constraint.. In *International Joint Conference on Artificial Intelligence*. 1240–1246.

[7] Xiaojun Chang, Feiping Nie, Yi Yang, Chengqi Zhang, and Heng Huang. 2016. Convex sparse PCA for unsupervised feature learning. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 11, 1 (2016), 3.

[8] Tom Croonenborghs, Kurt Driessens, and Maurice Bruynooghe. 2007. Learning relational options for inductive transfer in relational reinforcement learning. In *International Conference on Inductive Logic Programming*. Springer, 88–97.

[9] Manoranjan Dash and Huan Liu. 2000. Feature selection for clustering. In *Pacific-Asia Conference on knowledge Discovery and Data Mining*. Springer, 110–121.

[10] Hal Daume III. 2007. Frustratingly Easy Domain Adaptation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*. 256–263.

[11] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 248–255.

[12] Li Deng and Dong Yu. 2013. Deep learning for signal and information processing. *Microsoft Research Monograph* (2013).

[13] Paramveer S Dhillon and Lyle H Ungar. 2009. Transfer learning, feature selection and word sense disambguation. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*. Association for Computational Linguistics, 257–260.

[14] David L Donoho. 2000. High-dimensional data analysis: The curses and blessings of dimensionality. *AMS Math Challenges Lecture* (2000), 1–32.

[15] Liang Du and Yi-Dong Shen. 2015. Unsupervised feature selection with adaptive structure learning. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 209–218.

[16] Richard O Duda, Peter E Hart, and David G Stork. 1995. Pattern Classification and Scene Analysis. *ed: Wiley Interscience* (1995).

[17] Richard O Duda, Peter E Hart, and David G Stork. 2012. *Pattern Classification*. John Wiley & Sons.

[18] Mohamed Elhoseiny, Babak Saleh, and Ahmed Elgammal. 2013. Write a classifier: Zero-shot learning using purely textual descriptions. In *IEEE International Conference on Computer Vision*. 2584–2591.

[19] Florian Eyben, Martin Wöllmer, and Björn Schuller. 2010. Opensmile: the munich versatile and fast open-source audio feature extractor. In *Proceedings of the 18th ACM international conference on Multimedia*. ACM, 1459–1462.

[20] Ali Farhadi, Ian Endres, Derek Hoiem, and David Forsyth. 2009. Describing objects by their attributes. In *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 1778–1785.

[21] Fumiyo Fukumoto and Yoshimi Suzuki. 2015. Temporal-based feature selection and transfer learning for text categorization. In *Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K), 2015 7th International Joint Conference on*, Vol. 1. IEEE, 17–26.

[22] Yuchen Guo, Guiguang Ding, Yue Gao, and Jianmin Wang. 2016. Semi-supervised active learning with cross-class sample transfer. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*. 1526–1532.

[23] Yuchen Guo, Guiguang Ding, Xiaoming Jin, and Jianmin Wang. 2016. Transductive Zero-Shot Recognition via Shared Model Space Learning. In *Thirtieth AAAI Conference on Artificial Intelligence*.

[24] Isabelle Guyon and André Elisseeff. 2003. An introduction to variable and feature selection. *Journal of Machine Learning Research* 3, Mar (2003), 1157–1182.

[25] Xiaofei He, Deng Cai, and Partha Niyogi. 2005. Laplacian score for feature selection. In *Advances in Neural Information Processing Systems*. 507–514.

[26] Thibault Helleputte and Pierre Dupont. 2009. Feature selection by transfer learning with linear regularized models. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 533–547.

[27] Chenping Hou, Feiping Nie, Xuelong Li, Dongyun Yi, and Yi Wu. 2014. Joint embedding learning and sparse regression: A framework for unsupervised feature selection. *IEEE Transactions on Cybernetics* 44, 6 (2014), 793–804.

[28] Chenping Hou, Feiping Nie, Dongyun Yi, and Yi Wu. 2011. Feature selection via joint embedding learning and sparse regression. In *International Joint Conference on Artificial Intelligence*, Vol. 22. 1324.

[29] Eric H Huang, Richard Socher, Christopher D Manning, and Andrew Y Ng. 2012. Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*. Association for Computational Linguistics, 873–882.

[30] Dinesh Jayaraman and Kristen Grauman. 2014. Zero-shot recognition with unreliable attributes. In *Advances in Neural Information Processing Systems*. 3464–3472.

[31] Zhong Ji, Yuxin Sun, Yunlong Yu, Jichang Guo, and Yanwei Pang. 2018. Semantic softmax loss for zero-shot learning. *Neurocomputing* 316 (2018), 369–375.

[32] Gyeonghwan Kim and Venu Govindaraju. 1997. A lexicon driven approach to handwritten word recognition for real-time applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 19, 4 (1997), 366–379.

[33] Alex Krizhevsky and Geoffrey Hinton. 2009. *Learning multiple layers of features from tiny images*. Technical Report. University of Toronto.

[34] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*. 1097–1105.

[35] Christoph H Lampert, Hannes Nickisch, and Stefan Harmeling. 2009. Learning to detect unseen object classes by between-class attribute transfer. In *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 951–958.
[36] Neil D Lawrence and John C Platt. 2004. Learning to learn with the informative vector machine. In *Proceedings of the 21st International Conference on Machine learning*. ACM, 65.
[37] Daniel D Lee and H Sebastian Seung. 2001. Algorithms for non-negative matrix factorization. In *Advances in Neural Information Processing Systems*. 556–562.
[38] John A Lee and Michel Verleysen. 2007. *Nonlinear dimensionality reduction*. Springer Science & Business Media.
[39] Joon Ho Lee. 1997. Analyses of multiple evidence combination. In *ACM SIGIR Forum*, Vol. 31. ACM, 267–276.
[40] Joseph Lee Rodgers and W Alan Nicewander. 1988. Thirteen ways to look at the correlation coefficient. *The American Statistician* 42, 1 (1988), 59–66.
[41] Xin Li, Yuhong Guo, and Dale Schuurmans. 2015. Semi-supervised zero-shot classification with label representation learning. In *IEEE International Conference on Computer Vision*. 4211–4219.
[42] Yuanhong Li, Ming Dong, and Jing Hua. 2008. Localized feature selection for clustering. *Pattern Recognition Letters* 29, 1 (2008), 10–18.
[43] Zechao Li, Jing Liu, Yi Yang, Xiaofang Zhou, and Hanqing Lu. 2014. Clustering-guided sparse structural learning for unsupervised feature selection. *IEEE Transactions on Knowledge and Data Engineering* 26, 9 (2014), 2138–2150.
[44] Tong Lin and Hongbin Zha. 2008. Riemannian manifold learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 30, 5 (2008), 796–809.
[45] Huan Liu and Hiroshi Motoda. 1998. *Feature extraction, construction and selection: A data mining perspective*. Vol. 453. Springer Science & Business Media.
[46] Huan Liu and Hiroshi Motoda. 2007. *Computational methods of feature selection*. CRC Press.
[47] Xinwang Liu, Lei Wang, Jian Zhang, Jianping Yin, and Huan Liu. 2014. Global and local structure preservation for feature selection. *IEEE Transactions on Neural Networks and Learning Systems* 25, 6 (2014), 1083–1095.
[48] Yun Liu, Feiping Nie, Jigang Wu, and Lihui Chen. 2013. Efficient semi-supervised feature selection with noise insensitive trace ratio criterion. *Neurocomputing* 105 (2013), 12–18.
[49] David G Lowe. 1999. Object recognition from local scale-invariant features. In *IEEE International Conference on Computer Vision*, Vol. 2. IEEE, 1150–1157.
[50] Zhigang Ma, Feiping Nie, Yi Yang, Jasper RR Uijlings, Nicu Sebe, and Alexander G Hauptmann. 2012. Discriminating joint feature analysis for multimedia data understanding. *IEEE Transactions on Multimedia* 14, 6 (2012), 1662–1672.
[51] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *ICLR Workshop*.
[52] Feiping Nie, Heng Huang, Xiao Cai, and Chris H Ding. 2010. Efficient and robust feature selection via joint $l2$, 1-norms minimization. In *Advances in Neural Information Processing Systems*. 1813–1821.
[53] Feiping Nie, Shiming Xiang, Yangqing Jia, Changshui Zhang, and Shuicheng Yan. 2008. Trace Ratio Criterion for Feature Selection.. In *Proceedings of the 23rd National Conference on Artificial intelligenc*, Vol. 2. 671–676.
[54] Mohammad Norouzi, Tomas Mikolov, Samy Bengio, Yoram Singer, Jonathon Shlens, Andrea Frome, Greg S Corrado, and Jeffrey Dean. 2014. Zero-shot learning by convex combination of semantic embeddings. In *International Conference on Learning Representations*.
[55] 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.
[56] Sinno Jialin Pan and Qiang Yang. 2010. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering* 22, 10 (2010), 1345–1359.
[57] Genevieve Patterson and James Hays. 2012. Sun attribute database: Discovering, annotating, and recognizing scene attributes. In *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2751–2758.
[58] Vikas C Raykar, Balaji Krishnapuram, Jinbo Bi, Murat Dundar, and R Bharat Rao. 2008. Bayesian multiple instance learning: automatic feature selection and inductive transfer. In *Proceedings of the 25th International Conference on Machine learning*. ACM, 808–815.
[59] Marcus Rohrbach, Michael Stark, and Bernt Schiele. 2011. Evaluating knowledge transfer and zero-shot learning in a large-scale setting. In *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 1641–1648.
[60] Bernardino Romera-Paredes and PHS Torr. 2015. An embarrassingly simple approach to zero-shot learning. In *Proceedings of the 32nd International Conference on Machine Learning*. 2152–2161.
[61] Sam T Roweis and Lawrence K Saul. 2000. Nonlinear dimensionality reduction by locally linear embedding. *Science* 290, 5500 (2000), 2323–2326.
[62] Yvan Saeys, Iñaki Inza, and Pedro Larrañaga. 2007. A review of feature selection techniques in bioinformatics. *Bioinformatics* 23, 19 (2007), 2507–2517.
[63] Ruslan Salakhutdinov, Antonio Torralba, and Josh Tenenbaum. 2011. Learning to share visual appearance for multiclass object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 1481–1488.

[64] Matthias W Seeger. 2008. Bayesian inference and optimal design for the sparse linear model. *Journal of Machine Learning Research* 9, Apr (2008), 759–813.
[65] Cyrus Shaoul. 2010. The westbury lab wikipedia corpus. *Edmonton, AB: University of Alberta* (2010).
[66] Richard Socher, Milind Ganjoo, Christopher D Manning, and Andrew Ng. 2013. Zero-shot learning through cross-modal transfer. In *Advances in Neural Information Processing Systems*. 935–943.
[67] Mingkui Tan, Li Wang, and Ivor W Tsang. 2010. Learning sparse svm for feature selection on very high dimensional datasets. In *Proceedings of the 27th International Conference on Machine Learning*. 1047–1054.
[68] Jiliang Tang, Xia Hu, Huiji Gao, and Huan Liu. 2014. Discriminant Analysis for Unsupervised Feature Selection.. In *Proceedings of the 14th SIAM International Conference on Data Mining*. 938–946.
[69] Jiliang Tang and Huan Liu. 2012. Unsupervised feature selection for linked social media data. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 904–912.
[70] Robert Tibshirani. 1996. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)* 58, 1 (1996), 267–288.
[71] Selen Uguroglu and Jaime Carbonell. 2011. Feature selection for transfer learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 430–442.
[72] Ulrike Von Luxburg. 2007. A tutorial on spectral clustering. *Statistics and Computing* 17, 4 (2007), 395–416.
[73] De Wang, Feiping Nie, and Heng Huang. 2014. Unsupervised feature selection via unified trace ratio formulation and k-means clustering (TRACK). In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 306–321.
[74] Xiao-dong Wang, Rung-Ching Chen, Chao-qun Hong, Zhi-qiang Zeng, and Zhi-li Zhou. 2017. Semi-supervised multi-label feature selection via label correlation analysis with l1-norm graph embedding. *Image and Vision Computing* 63 (2017), 10–23.
[75] Zheng Wang, Xiaojun Ye, Chaokun Wang, Yuexin Wu, Changping Wang, and Kaiwen Liang. 2018. RSDNE: Exploring Relaxed Similarity and Dissimilarity from Completely-imbalanced Labels for Network Embedding. In *Thirty-Second AAAI Conference on Artificial Intelligence*.
[76] Gary M Weiss and Foster Provost. 2003. Learning when training data are costly: The effect of class distribution on tree induction. *Journal of Artificial Intelligence Research* 19 (2003), 315–354.
[77] Shuicheng Yan and Huan Wang. 2009. Semi-supervised learning by sparse representation. In *Proceedings of the 2009 SIAM International Conference on Data Mining*. SIAM, 792–801.
[78] Haiqin Yang, Michael R Lyu, and Irwin King. 2013. Efficient online learning for multitask feature selection. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 7, 2 (2013), 6.
[79] Yi Yang, Zhigang Ma, Alexander G Hauptmann, and Nicu Sebe. 2013. Feature selection for multimedia analysis by sharing information among multiple tasks. *IEEE Transactions on Multimedia* 15, 3 (2013), 661–669.
[80] Yiteng Zhai, Yew-Soon Ong, and Ivor W Tsang. 2014. The Emerging" Big Dimensionality". *IEEE Computational Intelligence Magazine* 9, 3 (2014), 14–26.
[81] Zheng Zhao and Huan Liu. 2007. Semi-supervised feature selection via spectral analysis. In *Proceedings of the 2007 SIAM International Conference on Data Mining*. SIAM, 641–646.
[82] Xiaojin Zhu. 2011. Semi-supervised learning. In *Encyclopedia of Machine Learning*. Springer, 892–897.
[83] Xiaojin Zhu, Zoubin Ghahramani, and John D Lafferty. 2003. Semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of the 20th International Conference on Machine learning*. 912–919.
[84] Yonghua Zhu, Xuejun Zhang, Rongyao Hu, and Guoqiu Wen. 2018. Adaptive structure learning for low-rank supervised feature selection. *Pattern Recognition Letters* 109 (2018), 89–96.