A Particle Swarm Optimization based Feature Selection Approach to Transfer Learning in Classification

Bach Hoai Nguyen, Bing Xue, and Peter Andreae
School of Engineering and Computer Science
Victoria University of Wellington, PO Box 600, Wellington 6140, New Zealand
{Hoai.Bach.Nguyen,Bing.Xue,Peter.Andreae}@ecs.vuw.ac.nz

ABSTRACT

Transfer learning aims to use acquired knowledge from existing (source) domains to improve learning performance on a different but similar (target) domains. Feature-based transfer learning builds a common feature space, which can minimize differences between source and target domains. However, most existing feature-based approaches usually build a common feature space with certain assumptions about the differences between domains. The number of common features needs to be predefined. In this work, we propose a new feature-based transfer learning method using particle swarm optimization (PSO), where a new fitness function is developed to guide PSO to automatically select a number of original features and shift source and target domains to be closer. Classification performance is used in the proposed fitness function to maintain the discriminative ability of selected features in both domains. The use of classification accuracy leads to a minimum number of model assumptions. The proposed algorithm is compared with four state-of-the-art feature-based transfer learning approaches on three well-known real-world problems. The results show that the proposed algorithm is able to extract less than half of the original features with better performance than using all features and outperforms the four benchmark semi-supervised and unsupervised algorithms. This is the first time Evolutionary Computation, especially PSO, is utilized to achieve feature selection for transfer learning.

CCS CONCEPTS

• **Computing methodologies** → **Feature selection**;

KEYWORDS

Transfer Learning, Domain Adaptation, Feature Selection, Classification, Particle Swarm Optimization

ACM Reference Format:
Bach Hoai Nguyen, Bing Xue, and Peter Andreae. 2018. A Particle Swarm Optimization based Feature Selection Approach to Transfer Learning in Classification. In *Proceedings of the Genetic and Evolutionary Computation Conference 2018 (GECCO '18),* Jennifer B. Sartor, Theo D'Hondt, and Wolfgang De Meuter (Eds.). ACM, New York, NY, USA, Article 4, 8 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

GECCO '18, July 15–19, 2018, Kyoto, Japan

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Machine learning techniques have been successfully applied to many real-world problems [27]. However, they usually require that training and test datasets are drawn from the same domain, i.e. same feature space and data distribution. If there is any change on the data distribution, most learning algorithms have to be re-trained using newly collected training data. In many real-world problems such as indoor Wifi localization [5], it is difficult and expensive to collect labeled data. It might be better to utilize acquired knowledge from the similar/related available labeled data (source domain) to improve learning performance on the unlabeled target data (target domain), which is the main motivation of transfer learning [18].

If the source and target domains have the same feature space, the transfer learning task can be further specified as *domain adaptation* [18], which is the focus of this work. The main task of domain adaptation is to reduce the differences between data distributions on the source and target domains. The most intuitive way is to find a good common feature representation to minimize the distributions' differences while preserving the discriminative ability in both source and target domains [1], which is known as *feature-based* domain adaptation. Once the common feature space is built, it does not need to re-train if more source data is available. Depends on the availability of labeled instances in the target domain, feature-based domain adaptation approaches can be divided into two categories: *semi-supervised approaches* with labeled instances and *unsupervised approaches* without labeled instances.

Many recent feature-based methods [7, 8, 15, 17, 21] attempt to build a new common latent feature space on which both source and target data can be projected, and traditional machine learning can be used to train a classifier to classify the projected target data. However, these methods usually have to assume models for transforming from the original feature space to the latent feature space. Furthermore, the dimensionality of the latent feature space must be pre-defined. Due to creating new high-level features, the interpretation of the constructed data is reduced and important information from the original features might be lost. Therefore, instead of building a new feature space, some other works [24, 25] select a number of features from the original features for reducing differences between distributions, so that the meaning of original features is maintained. Therefore, this work focuses on selecting original features known as feature selection for domain adaptation.

Although it has been shown that selecting invariant original features across different domains has good results, it is also important to select features with high discriminative ability (relevant features), which has not been paid enough attention in existing feature selection approaches for domain adaptation. Feature selection is not an easy task, especially when there is a large number of features.

Firstly, the search space increases exponentially with respect to the number of features. Secondly, the complex interactions between features makes the selection of an optimal feature subset harder. For example, selecting a number of weakly relevant features may significantly increase the classification performance while choosing only highly relevant features can result in redundant features [29]. Therefore, a global search is needed to achieve feature selection. Evolutionary Computation (EC) techniques have been widely applied to feature selection because of their potential global search ability. Among EC algorithms, Particle Swarm Optimization (PSO) is gaining more attention since it is simpler with fewer parameters in comparison with other EC techniques. Hence PSO is selected as a search mechanism to achieve feature selection in this work. This is the *first* time EC, especially PSO, is utilized to achieve feature selection for domain adaptation.

Most existing feature-based domain adaptation approaches make assumptions about *marginal distributions* (related to features) and/or *conditional distributions* (related to the class label) to simplify their models, which are easier to solve using numerical optimization techniques. Furthermore, although the two above distributions are considered, the discriminative ability on both domains is ignored in most works [13]. In comparison with feature selection for traditional machine learning, feature selection for domain adaptation is more challenging since it has to simultaneously maintain or improve the discriminative ability on both domains and minimize the differences between the two domains. In this work, we propose a new fitness function, which guides PSO to select a good feature subset with an expectation of leveraging differences in both marginal and conditional distributions while maintaining good accuracies on the source and target domains. The fitness function is designed with respect to the k-nearest neighbor (KNN) classification algorithm in order to minimize the number of model assumptions and lead to better accuracies on the target domain.

Goal: The overall goal of this paper is to develop a new fitness functions in PSO to achieve feature selection for domain adaptation. The fitness function aims to select invariant features containing relevant information about the class label, thereby results in high classification accuracies on the target domain. There will be three main components in the fitness function, which aim to reduce the classification error on the source domain, and the differences of both marginal and conditional distributions between two domains, respectively. Depends on whether there are labeled instances available in the target domain, the fitness function can adaptively change from its semi-supervised to unsupervised form. The proposed fitness function is examined on three real-world benchmark problems and compare with four sate-of-the-art traditional domain adaptation algorithms. Specifically, we will investigate:

- whether the proposed fitness function can assist PSO to select a good subset of features, which can achieve better performance than using all features on the target domain,
- whether the proposed PSO-based algorithm with the new fitness function can evolve common feature spaces, which achieve higher classification accuracy than the four traditional feature-based domain adaptation approaches, and
- whether the semi-supervised form can use class information on the target domain to outperform the unsupervised form.

2 BACKGROUND

2.1 Transfer Learning and Domain Adaptation

To understand transfer learning, it is necessary to explicitly define *domain* and *task*. A *domain* [18] consists of two components: feature space χ and a marginal probability distribution $P(X)$, where $X = \{x_1, \ldots, x_n\} \in \chi$. Two domains are different if there are differences between their feature spaces or marginal distributions. Given a domain $D(\chi, P(X))$, a *task*, denoted by $T(Y, f(.))$ [18], also consists of two components: a label space Y and a prediction function $f(.)$, which maps from the feature space χ to the label space Y. The prediction function $f(.)$, which can classify a new instance x, can be viewed as a conditional distribution $P(y|x)$. A *source domain data*, drawn from the source domain D_s, can be denoted as $Src = \{(x_{s_1}, y_{s_1}), (x_{s_2}, y_{s_2}), \ldots, (x_{m_s}, y_{m_s})\}$, where $x_{s_i} \in \chi$ is a source domain instance with the class label $y_{s_i} \in Y$ and m_s is the number of source instances. Similarly, a set of m_t target instances, called *target domain data*, drawn from the target domain D_t, is denoted by $Tar = \{(x_{t_1}, y_{t_1}), (x_{t_2}, y_{t_2}), \ldots, (x_{m_t}, y_{m_t})\}$. Given above concepts, Pan et al. [18] define transfer learning as below:

Transfer learning: *Given a source domain D_s and a learning task T_s, a target domain D_t and a learning task T_t, transfer learning aims to improve the learning of the target predictive function $f_t(.)$ in D_t using the knowledge in D_s and D_t where $D_s \neq D_t$ or $T_s \neq T_t$.*

Existing transfer learning works can be divided into four main categories, which are instance-based, clustering-based, parameter-based and feature-based transfer learning [18]. Instance-based transfer learning approaches assume that there are some parts of source data that are usable with a few labeled target instances to improve the target performance. Most works in this category aim to assign weights to source domain instances so that they can match the target domain well [19, 20]. Clustering-based approaches achieve transfer learning by building a similarity graph between all instances and the weight on each edge represents the similarity between two instances [6, 30]. Parameter-based transfer learning methods can be applied when the models to learn in the source and target domains have some parameters in common, such as support vector machine. The learned parameters from the source domain can be transfered to improve the target performance [4]. Feature-based transfer learning approaches aim to find a good feature representation, which simultaneously reduces the difference between the distributions on the source and target domains, and maintains important information of the source and target data. Not like other transfer learning approaches, feature-based transfer learning approaches usually do not require to re-train if more new instances from the source domain are obtained. Therefore, this work focus on developing feature-based transfer learning algorithms.

When the two domains have the same feature space i.e. $\chi_s = \chi_t$, transfer learning is specialized as domain adaptation. This work aims to achieve domain adaptation by building a good feature representation for both domains, which can be categorized as feature-based domain adaptation.

2.2 Feature-based Domain Adaptation

Most feature-based domain adaptation approaches aim to build a latent feature space as a bridge between different domains. The projected datasets of two original datasets on the new feature space are

expected to be closer than the original datasets. In [16], a dimensionality reduction method, which uses *Maximum Mean Discrepancy* [2] to measure the similarity between two distributions, is proposed to learn a new low-dimensional feature space. The new source dataset is then used to train a classifier which can be applied to classify the target instances. The work is further extended in [17] to address cases where information about the class label is available in the target domain. The two proposed algorithms, called TCA and STCA, are examined on two real-world applications, indoor WiFi localization and cross-domain text classification. The results show that both algorithms reduce the differences between source and target domains to achieve better target performance than using the original feature set. Shi and Sha [21] measure domain differences using mutual information between instances and their binary domain labels. The assumption is that the classes in two domains are well separated and instances across domains have to be close to each other if they belong to the same class. Shi and Sha show that by considering discriminative characteristics on both source and target domains, the classification performance is significantly improved over TCA/STCA [17]. However, the latent feature space is assumed to be a linear transformation of the original feature space. Yan et al. [31] proposed two algorithms, named MIDA and SMIDA, which can cope with continuous distributional changes in the feature space. The latent feature space is built to have a maximized independence with domain features. Some works project both source and target data into two different feature spaces and then build connections between the newly built spaces [3]. Although building a latent feature space may result in good performance, it loses meaning and possible important information of the original features.

An early work on feature selection for domain adaptation is performed by Uguroglu and Carbonel [25]. The task is to select original features that are not much different between two domains, which are called invariant features. Invariant and variant features can be distinguished based on their performance in minimizing domains' gap. It is shown that the proposed algorithm achieves better performance than TCA [17] on the digital recognition and Wifi localization problems. The efficiency of the work is improved by Tahmoresnezhad and Hashemi [24]. Particularly, the input dataset is split into k smaller sub-datasets. The proposed algorithm is run on all different sub-datasets, so each feature has k different weights. The average weight determines whether a feature is selected. However, in these works, a threshold value has to be pre-defined to select features. The features are selected individually based on their weights, which means that feature interactions are ignored. In addition, discriminative abilities on both source and target domains are not considered together, so some selected invariant features might be irrelevant to the class label. The above problems will be addressed in our work by developing a new fitness function in PSO, which can automatically determine the number of selected features. In the fitness function, classification performance is used to ensure the feature interactions are considered and the selected features have high discriminative abilities.

2.3 Particle Swarm Optimization

Particle Swarm Optimization (PSO) is a swarm intelligence algorithm proposed by Kennedy in 1995 [11], which is inspired by social behaviors of bird flocking. In oder to solve an optimization problem, PSO uses a swarm of particles to explore the search space parallelly. Each particle has its own position and velocity to move in the search space. During the searching process, they share their knowledge, i.e. their best positions with each other to guide the swarm towards the optimal solutions. The position and velocity are usually represented by numeric vectors, whose lengths are usually equal to the number of dimensions in the search space.

PSO is originally proposed to solve continuous problems, but it is also extended to cope with binary problems [12]. In comparison with continuous PSO, binary PSO (BPSO) is more suitable to solve feature selection since each binary entry can directly determine whether the corresponding feature is selected or not. However, standard BPSO's performance is limited in comparison with the continuous one [28]. The main reason is two continuous terms, *velocity* and *momentum*, are directly applied to binary search spaces without considering characteristics of binary movements. Nguyen et al. [14] re-define the two concepts to suit with binary search spaces and propose a novel BPSO algorithm, called sticky BPSO (SBPSO). Particularly, the velocity is considered as a probability vector in which each entry is the flipping probability of the corresponding position entry. The momentum is redefined as the tendency to stick with the current position, which is known as a stickiness (stk) property. Basically, if a particle has just flipped an entry, the entry's stk is set to the highest value of 1, so that the particle has enough time to explore around the entry's new value. In the following iterations, if the bit is not flipped, its stk is linearly decayed until 0. The following equation shows how stk of the d^{th} entry is updated.

$$stk_d^{t+1} = \begin{cases} 1, \textbf{ if } \text{the bit is just flipped} \\ \max(stk_d^t - \frac{1}{ustkS}, 0), \textbf{ otherwise} \end{cases} \quad (1)$$

where t is the t^{th} iteration and $ustkS$ is the pre-defined number of iterations to reduce stk from 1 to 0. The flipping probability p_d is then calculated by Eq. (2) based on stk_d, its personal best position, $pbest$, and its neighbor's best position, $gbest$.

$$p_d = i_s * (1 - stk_d) + i_p * |pbest_d - x_d| + i_g * |gbest_d - x_d| \quad (2)$$

where i_s, i_p and i_g control contributions of the three components. As suggested by [14], i_s, i_p, i_g and $ustkS$ are usually set to 0.1154, 0.4423, 0.4423 and 40, respectively. Eq. (3) shows how the d^{th} position entry is updated in SBPSO.

$$x_d^{t+1} = \begin{cases} 1 - x_d^t & \textbf{, if } rand() < p_d \\ x_d^t & \textbf{, otherwise} \end{cases} \quad (3)$$

It has been shown that SBPSO achieves good performance on feature selection [14]. Therefore, it is chosen as the search mechanism in this work.

In general, EC techniques, especially PSO, have been successfully applied to feature selection in traditional machine learning. This work is the first attempt to perform feature selection for domain adaptation using PSO to automatically choose a number of features. In order to achieve the goal, a novel fitness function is proposed to minimize the number of assumptions while still considering feature interactions and discriminabilities on both domains.

3 PROPOSED PSO-BASED FEATURE SELECTION FOR DOMAIN ADAPTATION

In this section, the proposed PSO based feature selection approach to domain adaptation is described. The main contribution is a new fitness function which allows the PSO-based feature selection algorithm to work in both cases: the class label information is available, i.e. semi-supervised, or not available on the target domain, i.e. unsupervised. To be convenient, the source dataset is named Src, the target un-labeled and labeled datasets are called $TarU$ and $TarL$, respectively.

3.1 New fitness function

In feature selection for domain adaptation, in order to achieve high classification performance on the target domain, the selected features must satisfy the following conditions: *1*) having a good discriminative ability on both domains, *2*) minimizing the difference between conditional distributions, and *3*) minimizing the difference between marginal distributions on source and target domains. Therefore, the fitness function has three components corresponding to the three above conditions, which can be seen in Eq. (4).

$$Fitness = sw * srcErr + tw * tarErr + stw * diffST \quad (4)$$

where $srcErr$ and $tarErr$ are classification errors on source and target data, which ensure the selected features to have a good discriminability on both domains (condition 1). Furthermore, $tarErr$ is obtained based on a classifier trained by the source data. Therefore, minimizing $tarErr$ leads to a smaller difference between conditional distributions on the two domains (condition 2). The last condition is achieved through $diffST$, which measures how different the two marginal distributions are (condition 3). The three terms will be explained in more details in the following sections. *sw, tw and stw* are used to control contributions of these three components and they sum up to 1, i.e. $sw + tw + stw$=1. The weight values can show the relationship between source and target domains, which will be illustrated in the parameter setting section.

The task of PSO is to search for feature subsets with the smallest fitness value. In this work, each particle is represented by an n-bit binary string where n is the total number of original features. Each bit corresponds to one feature and its value determines whether its corresponding feature is selected. Particularly, a feature is selected if and only if the corresponding bit value is 1.

3.2 Discriminability on the source domain

In Eq. (4), $srcErr$ is to ensure that the selected features have a high discriminative ability in the source domain. $srcErr$ is measured by the classification error rate. Many feature-based domain adaptation algorithms aim to minimize the differences between source and target domains while ignoring this important property on the source domain. The features obtained from these algorithms might not be useful if they cannot preserve the discriminative ability on the source domain and thereby on the target domain since the two domains become more similar under the selected features.

In order to ensure that $srcErr$ is not biased, $srcErr$ is calculated by applying 3-fold validation on the source dataset. Particularly, each fold plays the role of a test dataset one time while the other two folds are combined to form a training set. Eq. (5) shows how classification error rate is obtained on each fold.

$$ErrorRate = \frac{FP + FN}{TP + TN + FP + FN} \quad (5)$$

where TP, TN, FP and FN stand for true positive, true negative, false positive and false negative, respectively. The average value of three error rates on three folds is assigned to $srcErr$.

3.3 Discriminability on the target domain

In order to achieve a good discriminability on the target domain, we have to consider two possible situations: there are only a small number of labeled instances or there is no labeled instance, which form semi-supervised or unsupervised learning tasks, respectively.

3.3.1 Semi-supervised learning. If there is a small set of labeled instances available in the target domain, called $TarL$, $tarErr$ can be calculated as the classification error rate on $tarErr$ with Src being used as the training set. The reason is that if $tarErr$ is low, the two sets $TarL$ and Src are likely to have similar conditional distributions. Since $TarL$ and $TarU$ contain labeled and unlabeled instances drawn from the target domain, they should have the same conditional distribution. Therefore, minimizing $tarErr$ leads to leverage the distribution's difference between Src and $TarU$ as well. The $tarErr$ in the semi-supervised learning is named $tarErr_l$.

3.3.2 Unsupervised learning. In this case, we do not have labeled instances on the target domain, so the question is how to measure the discriminability without using labels. Based on the idea that the two closest instances usually belong to the same class, we can *estimate* the classification error on the target domain by using labeled source domain instances, given a set of selected features. In particular, suppose that x_{t_1} and x_{t_2} are two closest unlabeled instances on the target domain ($TarU$), they are very likely to have the same class label. Since the class labels of x_{t_1} and x_{t_2} are not available, we can *estimate* their class labels based on their two closest instances from the source domain, x_{s_1} and x_{s_2}. If x_{s_1} and x_{s_2} are in the same class, x_{t_1} and x_{t_2} are also in the same class, which means the selected features are good for grouping similar instances into the same class. Otherwise, x_{t_1} and x_{t_2} are in different classes indicating the poor disciminability of the selected features. The $tarErr$ in the unsupervised learning, called $tarErr_u$, is the division between the number of closest target instances being estimated in *different* classes, and the total number of instances in $TarU$. Note that since the number of closest instance pairs equal to the number instances in $TarU$, $tarErr_u$ is in the range [0,1]. Details on calculating $tarErr_u$ are shown in Algorithm 1.

On both semi-supervised and unsupervised learning cases, the target of $tarErr$ is to ensure that if the selected features has a good discriminative ability on the source domain, they should also have a high discriminability on the target domain. Conceptually, this is similar to the idea of making conditional distributions similar across domains in existing feature-based domain adaptation approaches. However, the use of $tarErr$ does not require any model assumption about the conditional distribution and it works closely with the KNN classification algorithm, which is expected to result in a high classification performance.

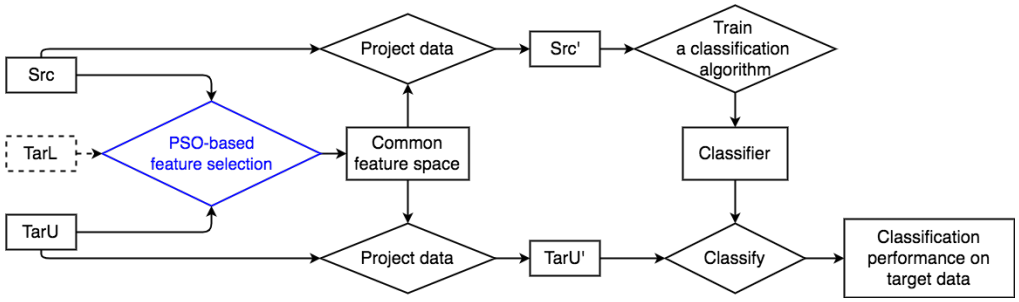

Figure 1: PSO-based Feature Selection for Domain Adaptation

Algorithm 1 : Calculate $tarErr_u$

```
correct =0
for each instance x_{t_i} in TarU do
    find its closest instance in the source domain, x_{s_i}
    find its closest instance in the target domain, x_{t_j}
    find the closest instance of x_{t_j} in the source domain, x_{s_j}
    if x_{s_i} and x_{s_j} are in the same class then
        correct = correct +1
    end if
end for
tarErr_u = 1 - correct/|TarU|
```

3.4 Difference between marginal distributions

The last term, $diffST$, in Eq. (4) aims to minimize the difference between the marginal distributions. Maximum Mean Discrepancy (MMD) [2] is used to measure the difference between the two marginal distributions, as shown in Eq. (6). This metric is widely used in many feature-based approaches.

$$D(Src, TarU) = \left\| \frac{1}{|Src|} \sum_{i=1}^{|Src|} \phi(Src_i) - \frac{1}{|TarU|} \sum_{i=1}^{|TarU|} \phi(TarU_i) \right\|_H \tag{6}$$

where $\phi(x) : X \rightarrow H$, H is a universal reproducing kernel Hilbert space. Note that in both semi-supervised and unsupervised learning, $TarU$ is always used in the Eq. (6) since the final task is to well classify instances from $TarU$. Using the kernel trick, i.e. $k(z_i, z_j^T) = \phi(z_i)\phi(z_j^T)$, where k is a positive definite kernel [17], Eq. (6) can be rewritten as:

$$\begin{aligned} D(Src, TarU) = \Bigg(& \frac{1}{|Src|^2} \sum_{i=1}^{|Src|} \sum_{j=1}^{|Src|} k(Src_i, Src_j) \\ & + \frac{1}{|TarU|^2} \sum_{i=1}^{|TarU|} \sum_{j=1}^{|TarU|} k(TarU_i, TarU_j) \\ & - \frac{2}{|Src||TarU|} \sum_{i=1}^{|Src|} \sum_{j=1}^{|TarU|} k(Src_i, TarU_j) \Bigg)^{1/2} \end{aligned} \tag{7}$$

According to [23], Gaussian Radial Basis Function [RBF, $k(x, y) = exp(||x-y||^2/2\sigma^2)$] is able to detect more types of dependence than linear or polynomial kernels. Thus RBF is used in this work. Its kernel width (σ) is automatically selected for each case based on the "median trick" [22].

3.5 Overall Algorithm

The overall structure of the proposed system is shown in Figure 1. The main contribution of this work is the PSO-based feature selection algorithm, which is marked in black. In general, source (Src), unlabeled target ($TarU$) and possibly labeled target ($TarL$) data are used to evaluate particles using the proposed fitness function, Eq. (4). Based on the final feature subset selected by PSO, both Src and $TarU$ are projected on the common feature space to form two new data, Src' and $TarU'$, which should share the same data distributions. The KNN classification algorithm (k=1) uses Src' as the training set to classify instances in $TarU'$ to obtain the classification performance on the target domain. Depends on whether $TarL$ is available in the target domain, either $tarErr_l$ or $tarErr_u$ is used in Eq. (4). We name the two algorithms using $tarErr_l$, $tarErr_u$ as SemPSO, UnPSO, respectively.

4 EXPERIMENT DESIGN

The proposed two algorithms are compared with using all features, two well-known unsupervised feature-based domain adaptation algorithms, TCA [17], MIDA [31], and their extended semi-supervised algorithms, STCA [17], SMIDA [31]. The systems of the four benchmark traditional algorithms are similar to the one shown in Figure 1, except for the step building the common feature space.

4.1 Benchmark Datasets

All the algorithms are examined on three well-known real-world problems, Gas Sensor [26], Handwritten Digits [24] and Object Recognition [8, 10], which are separated by dashed lines as shown in Table 1, where #C and #F represent the number of classes and features. Each problem contains many cases, which have the same number of classes and features, but might have different numbers of instances in the source (Src) and target domains ($TarU, TarL$).

The gas sensor array drift datasets are collected by Vergara et al. [26] using 16 gas sensors over 36 months. The task is to classify instances into six different kinds of gas. The whole datasets are divided into 10 batches according to the acquisition time. The 1^{st} batch is used as the source dataset and each batch from the 2^{nd} to the 10^{th} ones are used as the target dataset, which forms 9 domain adaptation cases.

USPS and MNIST [24] are two handwritten digit datasets, which share 10 classes of digits. The USPS dataset is collected by scanning envelops from US Postal Service while MNIST is taken from mixed American Census Bureau employees and American high school students, so they have very different distributions. In this problem,

Table 1: Domain adaptation problems.

Problem	Cases	#C	#F	|Src|	|TarU|	|TarL|
Gas Sensor	1-2	6	129	178	746	498
	1-3	6	129	178	951	635
	1-4	6	129	178	97	64
	1-5	6	129	178	118	79
	1-6	6	129	178	1380	920
	1-7	6	129	178	2168	1445
	1-8	6	129	178	176	118
	1-9	6	129	178	282	188
	1-10	6	129	178	2160	1440
Handwritten Digits	MNIST-USPS	10	257	800	1080	720
	USPS-MNIST	10	257	720	1200	800
Object Recognition	A-C	10	801	384	674	449
	A-D	10	801	384	94	63
	A-W	10	801	384	177	118
	C-A	10	801	449	574	384
	C-D	10	801	449	94	63
	C-W	10	801	449	177	118
	D-A	10	801	63	574	384
	D-C	10	801	63	674	449
	D-W	10	801	63	177	118
	W-A	10	801	118	574	384
	W-C	10	801	118	674	449
	W-D	10	801	118	94	63

there are two domain adaptation cases, in which either MNIST or USPS is the source dataset and the other one is the target dataset.

The last problem is to recognize 10 objects from four different image sources including Caltech-256 (C) [10], Amazon (A), Webcam (W) and DSLR (D) [8]. To form a domain adaptation case in this problem, we select one image source as the source domain and another image source as the target domain. Therefore, there are 12 cases for the object recognition problem.

In general, there are 23 domain adaptation cases, which have different numbers of classes, features or different numbers of instances in the source and target domain.

4.2 Parameter Settings

Parameters of the four traditional algorithms (TCA, STCA, MIDA and SMIDA), including their kernel and kernel widths, are tuned for the best accuracy using a heuristic search [31] on each case. The parameters of SBPSO are set according to recommendations in its original paper [14], as described in Section 2.3. Each PSO-based algorithm is run 30 independent times on each case.

To tune the three weights in Eq. (4), three cases with the lowest classification accuracy from each problem are selected as representatives, which are "1-8", "USPS-MNIST" and "C-W". Different values of the three weights are examined by running SemPSO one time on each selected case. The evolved feature subsets are used to obtain two projected datasets of Src and $TarL$, called Src' and $TarL'$. The best setting for each problem is selected according to the best and the average classification accuracy on $TarL'$. Particularly, on Gas Sensor, sw, tw and stw are set to 0.1, 0.9 and 0.0, respectively. On Handwrittend Digits and Object Recognition, the three values are (0.1, 0.7, 0.2) and (0.0, 0.1, 0.9), correspondingly. Based on the parameters, it can be seen that different problems have different relationships between source and target domains. For example, on Gas Sensor, tw is large which means that the main difference between the two domains is the conditional distribution. Thus tuning the weights may reveal the relationship between domains, which cannot be easily achieved by most traditional approaches. The setting of UnPSO mainly follows SemPSO, except for Object Recognition, where sw is set to 0. Since $TarL$ is not available in UnPSO, $srcErr$ seems to be more reliable than $tarErr$, so sw is set to 0.1 instead of 0.0.

5 RESULTS

The results are shown in Table 2, where #F and Acc represent for the number of selected features and the average classification performance. The number of features selected by PSO are used as the pre-defined number of features for four traditional methods to ensure a relatively fair comparison. In the table, the best accuracies are marked in bold while the second best ones are underlined. A significance Wilcoxon test with significance level set to 0.05 is used to compare between semi-supervised, unsupervised approaches to examine the effect of the proposed fitness function in both learning cases. Particularly, SemPSO is compared with Full/STCA/SMIDA in Table 3. Table 4 shows the comparisons between UnPSO and Full/TCA/MIDA. In each table cell, the three numbers represent the number of cases that the PSO-based approaches are significantly better, similar or worse than the other benchmark algorithms.

5.1 SemPSO/UnPSO vs using all features

As can be seen from Table 2, on all cases, SemPSO achieves significantly better performance than using all features. For example on Gas Sensor, SemPSO usually improves 20% over the original feature set, especially on the 1-8 case, the accuracy of SemPSO is almost three times better. In the 1-6 case, the accuracy of using all features are already quite high, which means that the two domains are very similar and most features are invariant. However, SemPSO still manages to improve the accuracy by 5%. The possible reason is the classification accuracies in Eq. (4) assist PSO to remove irrelevant or redundant features from both domains.

Similar to SemPSO, UnPSO also achieves good performance despite of lacking information about the class label in the target domain. Table 4 shows that UnPSO is worse than the original feature sets on only two cases while being significantly better on 17 out of the 23 cases. Although on 1-10 and D-A, UnPSO's accuracies are at most 0.9% less than using all features, it only selects less than a half number of the features. On the other hand, the largest improvement by UnPSO is on W-C, where UnPSO evolves a set of features which are almost three times more accurate than using all features.

The experimental results show that PSO guided by the proposed fitness function can automatically reduce half of the numbers of features and achieve better classification performance than using all features. The fitness function not only selects invariant features but also extracts relevant ones to improve the classification accuracy.

5.2 SemPSO vs STCA/SMIDA

As can be seen in Table 3, SemPSO well utilizes the class label information on the target domain to significantly improve the classification performance. On all cases, SemPSO significantly outperforms

Table 2: Overall results on 23 cases.

Cases	Full		Unsupervised				Semi-supervised			
			TCA	MIDA	UnPSO		STCA	SMIDA	SemPSO	
	#F	Acc	Acc	Acc	#F	Acc	Acc	Acc	#F	Acc
1-2	128	69.57	63.40	61.80	58.63	76.71	72.39	68.50	49.33	**90.90**
1-3	128	70.24	60.04	63.41	57.00	72.66	72.24	63.83	44.30	**94.58**
1-4	128	61.86	52.58	56.70	62.40	61.55	68.04	58.76	55.23	**81.34**
1-5	128	70.34	49.15	75.42	53.70	71.89	66.95	75.42	56.73	**76.44**
1-6	128	89.64	79.06	80.14	59.00	89.56	84.35	80.58	52.47	**94.53**
1-7	128	53.60	57.84	56.83	55.20	58.09	63.65	55.81	45.73	**71.54**
1-8	128	26.70	12.50	29.55	60.33	32.41	9.09	13.64	41.27	**79.47**
1-9	128	46.45	21.28	18.44	57.93	53.01	29.79	**71.63**	47.67	67.86
1-10	128	49.12	49.54	47.31	58.83	48.19	57.73	45.28	36.77	**68.35**
MNIST-USPS	256	59.63	35.65	34.91	113.33	65.90	55.65	35.93	104.23	**72.54**
USPS-MNIST	256	23.83	20.83	21.17	74.13	49.82	10.67	19.92	97.33	**56.14**
A-C	800	22.55	27.89	28.19	414.83	27.24	19.44	**31.45**	410.20	30.53
A-D	800	18.09	22.34	23.40	409.17	26.74	17.02	26.60	392.30	**29.61**
A-W	800	22.03	26.55	27.12	412.90	28.31	20.90	32.20	401.73	**35.37**
C-A	800	24.39	30.84	30.49	396.23	28.82	24.56	15.33	394.03	**33.05**
C-D	800	22.34	6.38	**26.60**	390.47	25.53	23.40	25.53	383.30	26.06
C-W	800	16.95	20.90	20.34	396.03	25.12	20.34	23.73	389.03	**28.44**
D-A	800	22.65	17.94	17.42	397.63	22.10	11.50	23.34	394.00	**31.40**
D-C	800	23.74	24.78	23.44	379.83	22.38	12.76	26.26	393.80	**29.06**
D-W	800	41.24	10.17	18.08	396.10	47.18	42.94	18.08	405.07	**51.94**
W-A	800	21.95	9.58	24.91	403.67	21.62	10.63	24.22	401.43	**30.21**
W-D	800	44.68	13.83	14.89	407.57	**52.27**	7.45	18.09	403.17	50.32
W-C	800	8.00	14.24	14.09	412.17	21.39	14.39	10.83	391.27	**27.72**

Table 3: SemPSO vs semi-supervised methods

Full	STCA	SMIDA
23/0/0	23/0/0	20/1/2

Table 4: UnPSO vs unsupervised methods

Full	TCA	MIDA
17/4/2	18/1/4	17/2/4

STCA. In comparison with SMIDA, SemPSO is significantly better on 20 out of the 23 cases. On the 1-6 case, where the two domains are very similar, both STCA and SMIDA build new latent feature spaces, which perform worse than the original features. In this case, the important information of original features is discarded by the two traditional methods. On the other hand, SemPSO aims to select relevant original features on both domains, so it can improve the performance over using all features. The two traditional methods aim to maximize the dependence between the features and labels on the target domain. However, the dependency is implicitly optimized through the Hilbert-Schmidt Independence Criterion (HSIC) [9]. Meanwhile, in our proposed fitness function, the classification performance is used to explicitly presents the dependence between the features and labels, which leads to higher accuracies on the target model. In comparison with UnPSO, except for W-D, SemPSO achieves better classification performance on all other cases. The results show that the actual classification error on *TarL* works better than estimating the classification error on *TarU*.

5.3 UnPSO vs TCA/MIDA

The significance test results between UnPSO and TCA/MIDA are shown in Table 4. As shown in the table, UnPSO is similar to the two traditional algorithms only on at most 2 cases while being significantly better on at least 17 cases. On the Gas Sensor problem, UnPSO usually achieves 10% accuracy better than at least one of the two traditional methods. Especially on the difficult case 1-9, UnPSO is at least three times more accurate than TCA and MIDA. UnPSO also outperforms the two traditional methods on the two handwriting digital cases. Only on the 12 object recognition cases, TCA and MIDA can achieve comparable performance in comparison with UnPSO. UnPSO is similar or better than the other two methods on only 9 out of the 12 cases. The possible reason is the estimation process of $tarErr_u$ mainly bases on Euclidean distances, which may not work well on such high-dimensional datasets (800 features).

5.4 Overall Comparisons

As can be seen in Table 2, in general, SemPSO achieves the best classification accuracy on 19 out of the 23 cases while being ranked as the second best algorithm on the other four cases. UnPSO evolves the best common feature set on one case and acquires the second best accuracy on nine cases. The achievement of UnPSO is much better than the best traditional algorithm, SMIDA, which obtains the best or second best performance on 6 cases. Note that in the overall comparisons, UnPSO is also compared with semi-supervised algorithms like SMIDA, which assume some instances are labeled in the target domain. Therefore, the outperformance of UpPSO to

other semi-supervised traditional algorithms suggests that the estimation of target classification error based on the source domain can improve the discriminative ability on the target domain and reduce the differences between their conditional distributions, which are assumed to be the same in the four traditional approaches. The classification error rate on the source domain also plays an important role in the fitness function. Normally, traditional feature-based adaptation approaches focus only on producing invariant features, while the source classification performance ensures that the selected features have high discriminative abilities on both domains when they become similar. In terms of computation time, the PSO-based algorithms are more computationally intensive than the four traditional approaches since they involve classification processes in the evaluation step.

6 CONCLUSIONS AND FUTURE WORK

In this work, a novel fitness function is developed to assist PSO to automatically select a subset of original features, which can improve classification performance in domain adaptation problems. The proposed fitness function aims to select relevant and invariant features across different domains. The fitness function can flexibly adapt with unsupervised or semi-supervised domain adaptation, depends on the availability of labels on the target domain. Based on that, two PSO-based feature selection algorithms for domain adaptation are proposed and examined on three well-known real-world problems containing 23 domain adaptation cases, in total. The proposed algorithm, called UnPSO and SemPSO, are compared with four traditional feature-based domain adaptation algorithms, TCA, STCA, MIDA and SMIDA. The results suggest that explicitly presenting feature and label relations by classification performance results in better common feature spaces than relying on model assumptions. Furthermore, incorporating the discriminability on the source domain in the fitness function guides PSO to search for feature subsets with a high discriminative ability on both domains.

This is the first work utilizing EC, specifically PSO, to achieve feature-based domain adaptation. In the future, we will further investigate their potential to achieve even better performance. For example, the difference between two marginal distributions is calculated based on the MMD model, which may not perform well on the high-dimensional dataset. We will work on estimating the difference without any model assumption. The three weights in the fitness function can be adaptively changed, by analyzing the relationships between source and target domains.

REFERENCES

[1] Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. 2007. Analysis of representations for domain adaptation. In *Advances in neural information processing systems*. 137–144.

[2] Karsten M Borgwardt, Arthur Gretton, Malte J Rasch, Hans-Peter Kriegel, Bernhard Schölkopf, and Alex J Smola. 2006. Integrating structured biological data by kernel maximum mean discrepancy. *Bioinformatics* 22, 14 (2006), e49–e57.

[3] Zhen Cui, Wen Li, Dong Xu, Shiguang Shan, Xilin Chen, and Xuelong Li. 2014. Flowing on Riemannian manifold: Domain adaptation by shifting covariance. *IEEE Transactions on Cybernetics* 44, 12 (2014), 2264–2273.

[4] Theodoros Evgeniou and Massimiliano Pontil. 2004. Regularized multi–task learning. In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 109–117.

[5] Shih-Hau Fang and Tsung-Nan Lin. 2008. Indoor location system based on discriminant-adaptive neural network in IEEE 802.11 environments. *IEEE Transactions on Neural Networks* 19, 11 (2008), 1973–1978.

[6] Jing Gao, Wei Fan, Jing Jiang, and Jiawei Han. 2008. Knowledge transfer via multiple model local structure mapping. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 283–291.

[7] Boqing Gong, Yuan Shi, Fei Sha, and Kristen Grauman. 2012. Geodesic flow kernel for unsupervised domain adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition*. 2066–2073.

[8] Raghuraman Gopalan, Ruonan Li, and Rama Chellappa. 2011. Domain adaptation for object recognition: An unsupervised approach. In *IEEE International Conference on Computer Vision*. 999–1006.

[9] Arthur Gretton, Olivier Bousquet, Alex Smola, and Bernhard Schölkopf. 2005. Measuring statistical dependence with Hilbert-Schmidt norms. In *International Conference on Algorithmic Learning Theory*. Springer, 63–77.

[10] Gregory Griffin, Alex Holub, and Pietro Perona. 2007. Caltech-256 object category dataset. (2007).

[11] James Kennedy. 2011. Particle swarm optimization. In *Encyclopedia of machine learning*. Springer, 760–766.

[12] James Kennedy and Russell C Eberhart. 1997. A discrete binary version of the particle swarm algorithm. In *IEEE International Conference on Systems, Man, and Cybernetics*, Vol. 5. 4104–4108.

[13] Mingsheng Long, Jianmin Wang, Guiguang Ding, Jiaguang Sun, and S Yu Philip. 2013. Transfer feature learning with joint distribution adaptation. In *IEEE International Conference on Computer Vision*. 2200–2207.

[14] Bach Hoai Nguyen, Bing Xue, and Peter Andreae. 2017. A novel binary particle swarm optimization algorithm and its applications on knapsack and feature selection problems. In *Intelligent and Evolutionary Systems*. Springer, 319–332.

[15] Maxime Oquab, Leon Bottou, Ivan Laptev, and Josef Sivic. 2014. Learning and transferring mid-level image representations using convolutional neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition*. 1717–1724.

[16] Sinno Jialin Pan, James T. Kwok, and Qiang Yang. 2008. Transfer Learning via Dimensionality Reduction. In *Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 2*. 677–682.

[17] 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.

[18] Sinno Jialin Pan and Qiang Yang. 2010. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering* 22, 10 (2010), 1345–1359.

[19] Weike Pan, Hao Zhong, Congfu Xu, and Zhong Ming. 2015. Adaptive Bayesian personalized ranking for heterogeneous implicit feedbacks. *Knowledge-Based Systems* 73 (2015), 173–180.

[20] David Pardoe and Peter Stone. 2010. Boosting for regression transfer. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*. Omnipress, 863–870.

[21] Yuan Shi and Fei Sha. 2012. Information-theoretical Learning of Discriminative Clusters for Unsupervised Domain Adaptation. In *Proceedings of the 29th International Coference on International Conference on Machine Learning (ICML'12)*. 1275–1282.

[22] Le Song, Byron Boots, Sajid M Siddiqi, Geoffrey J Gordon, and Alex Smola. 2010. Hilbert space embeddings of hidden Markov models. (2010).

[23] Le Song, Alex Smola, Arthur Gretton, Justin Bedo, and Karsten Borgwardt. 2012. Feature selection via dependence maximization. *Journal of Machine Learning Research* 13, May (2012), 1393–1434.

[24] Jafar Tahmoresnezhad and Sattar Hashemi. 2016. An Efficient yet Effective Random Partitioning and Feature Weighting Approach for Transfer Learning. *International Journal of Pattern Recognition and Artificial Intelligence* 30, 02 (2016), 1651003.

[25] 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.

[26] Alexander Vergara, Shankar Vembu, Tuba Ayhan, Margaret A Ryan, Margie L Homer, and Ramón Huerta. 2012. Chemical gas sensor drift compensation using classifier ensembles. *Sensors and Actuators B: Chemical* 166 (2012), 320–329.

[27] Xindong Wu, Vipin Kumar, J Ross Quinlan, Joydeep Ghosh, Qiang Yang, Hiroshi Motoda, Geoffrey J McLachlan, Angus Ng, Bing Liu, S Yu Philip, et al. 2008. Top 10 algorithms in data mining. *Knowledge and Information Systems* 14, 1 (2008), 1–37.

[28] Bing Xue, Su Nguyen, and Mengjie Zhang. 2014. A new binary particle swarm optimisation algorithm for feature selection. In *European Conference on the Applications of Evolutionary Computation*. Springer, 501–513.

[29] Bing Xue, Mengjie Zhang, Will N Browne, and Xin Yao. 2016. A survey on evolutionary computation approaches to feature selection. *IEEE Transactions on Evolutionary Computation* 20, 4 (2016), 606–626.

[30] Gui-Rong Xue, Wenyuan Dai, Qiang Yang, and Yong Yu. 2008. Topic-bridged PLSA for cross-domain text classification. In *Proceedings of the 31st International ACM SIGIR Conference on Research and Development in Information Retrieval*. 627–634.

[31] K. Yan, L. Kou, and D. Zhang. 2018. Learning Domain-Invariant Subspace Using Domain Features and Independence Maximization. *IEEE Transactions on Cybernetics* 48, 1 (2018), 288–299.