Domain Adaptation Transfer Extreme Learning Machines

Lei Zhang[1,2] and David Zhang[2]

[1] College of Computer Science, Chongqing University, No.174 Shazheng Street, ShapingBa District, Chongqing, China
leizhang@cqu.edu.cn
[2] Department of Computing, The Hong Kong Polytechnic Unviersity, Hung Hom, Hong Kong
csdzhang@comp.polyu.edu.hk

Abstract. Extreme learning machines (ELMs) have been confirmed to be efficient and effective learning techniques for pattern recognition and regression. However, ELMs primarily focus on the supervised, semi-supervised and unsupervised learning problems in single domain and the generalization ability in multiple domains based learning issues is hardly studied. This paper aims to propose a unified framework of ELMs with domain adaptation and improve their transfer learning capability in cross domains without loss of the computational efficiency of traditional ELMs. We integrate domain adaptation into ELMs and two algorithms including source domain adaptation transfer ELM (TELM-SDA) and target domain adaptation transfer ELM (TELM-TDA) are proposed. For insight of the difference among ELM, TELM-SDA and TELM-TDA, two *remarks* are provided. Experiments on the popular sensor drift big data with multiple batches in machine olfaction, the results clearly demonstrate the characteristics of the proposed domain adaptation transfer ELMs that they can not only copy with sensor drift efficiently without cumbersome measures comparable to state-of-the-art methods but also bring new perspectives for ELM.

Keywords: Extreme learning machine, domain adaptation, transfer learning, semi-supervised learning.

1 Introduction

Extreme learning machine (ELM), proposed for solving a single layer feed-forward network (SLFN) by Huang et al [1, 2], has been proven to be effective and efficient algorithms for pattern classification and regression in different fields. ELM can analytically determine the output weights between the hidden layer and output layer using Moore-Penrose generalized inverse by adopting the square loss of prediction error, which then turns into solving a regularized least square problem efficiently in closed form. The output of the hidden layer is activated by an infinitely differentiable function with randomly selected input weights and biases of the hidden layer. Huang [3] rigorously prove that the input weights and hidden layer biases can be randomly

J. Cao et al. (eds.), *Proceedings of ELM-2014 Volume 1*,
Proceedings in Adaptation, Learning and Optimization 3, DOI: 10.1007/978-3-319-14063-6_10

assigned if the activation function is infinitely differentiable, who also showed that single SLFN with randomly generated additive or RBF nodes with such activation functions can universally approximate any continuous function on any compact subspace of Euclidean space [4].

In recent years, ELM has witnessed a number of improved versions in models, algorithms and real-world applications. ELM shows a comparable or even higher prediction accuracy than that of SVMs which solves a quadratic programming problem. In [3], their differences have been discussed. Some specific examples of improved ELMs have been listed as follows. As the output weights are computed with prefixed input weights and biases of hidden layer, a set of non-optimal input weights and hidden biases may exist. Additionally, ELM may require more hidden neurons than conventional learning algorithms in some special applications. Therefore, Zhu et al [5] proposed an evolutionary ELM for more compact networks that speed the response of trained networks. In terms of the imbalanced number of classes, a weighted ELM was proposed for binary/multiclass classification tasks with both balanced and imbalanced data distribution [6]. Due to that the solution of ELM is dense which will require longer time for training in large scale applications, Bai et al [7] proposed a sparse ELM for reducing storage space and testing time. Besides, Li et al [8] also proposed a fast sparse approximation of ELM for sparse classifiers training at a rather low complexity without reducing the generalization performance. For all the versions of ELM mentioned above, supervised learning framework was widely explored in application which limits its ability due to the difficulty in obtaining the labeled data. Therefore, Huang et al [9] proposed a semi-supervised ELM, in which a manifold regularization with graph Laplacian was set, and under the formulation of semi-supervised ELM, an unsupervised ELM was also explored.

In the past years, the contributions to ELM theories and applications have been made substantially by researchers from various fields. However, with the rising of big data, the data distribution obtained in different stages with different experimental conditions may change, i.e. from different domains. It is also well know that collection of labeled instances is tedious and labor ineffective, while the classifiers trained by a small number of labeled data are not robust and therefore lead to weak generalization, especially for large-scale application. Though ELM performs better generalization in a number of labeled data, the transferring capability of ELM may be reduced with very little number of labeled training instances from different domains. Domain adaptation methods have been proposed for classifiers learning with a few labeled instances from target domains by leveraging a number of labeled samples from the source domains [10-14]. Domain adaptation methods have also been employed for object recognition and sensor drift compensation [15, 16]. It is worth noting that domain adaptation is different from semi-supervised learning which assumes that the labeled and unlabeled data are from the same domain in classifier training.

In this paper, we extend ELMs to handle domain adaptation problems for improving the transferring capability of ELM between multiple domains with very few labeled guide instances in target domain, and overcome the generalization disadvantages of ELM in multi-domains application. Inspired by domain adaptation theory, two domain adaptation ELMs with similar structures but different knowledge adaptation characteristics are proposed for domain adaptation learning and knowledge transfer. The proposed domain adaptation ELMs are named as source domain adaptation transfer ELM (TELM-SDA) and target domain adaptation transfer ELM (TELM-TDA), respectively. Specifically, TELM-TDA learns a classifier using the very few labeled instances from target domain, while the remaining numerous unlabeled data are also fully exploited by approximating the prediction of the base classifier which can be trained in the source domain by regularized ELM or SVM. TELM-SDA is a more instinct framework which learns a classifier by using the large number of labeled data from the source domain, and very few labeled instances from target domain as regularization. From the learning mechanism of both domain adaptation ELMs, TELM-TDA has larger computation than TELM-SDA, due to the base classifier training and the numerous unlabeled data from target domain considered in learning. It is worth noting that both TELM-TDA and TELM-SDA can be formed into a unified ELM framework which refers to two stages including random feature mapping and output weights training.

The rest of this paper is organized as follows. In Section 2, a brief review of ELM is presented. In Section 3, the proposed TELM-SDA is illustrated in principle and algorithm. In Section 4, the proposed TELM-TDA is presented with its principle and algorithm. In Section 5, we present the experiments and results on the popular sensor drift data with multiple batches collected by electronic nose with 3 years for gas recognition. The conclusion of this paper is given in Section 6.

2 Related Work: A Brief Review of ELM

Given N samples $[\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N]$ and their corresponding target$[\mathbf{t}_1, \mathbf{t}_2, \cdots, \mathbf{t}_N]$, where $\mathbf{x}_i = [x_{i1}, x_{i1}, \cdots, x_{in}]^{\mathrm{T}} \in \mathbb{R}^n$ and $\mathbf{t}_i = [t_{i1}, t_{i1}, \cdots, t_{im}]^{\mathrm{T}} \in \mathbb{R}^m$. The output of the hidden layer is denoted as $h(\mathbf{x}_i) \in \mathbb{R}^{1\times L}$, where L is the number of hidden nodes and $h(\cdot)$ is the activation function (e.g. RBF function, sigmoid function). The output weights between the hidden layer and the output layer being learned is denoted as $\boldsymbol{\beta} \in \mathbb{R}^{L\times m}$, where O is the number of output nodes.

Regularized ELM aims to solve the output weights by minimizing the squared loss summation of prediction errors and the norm of the output weights for over-fitting control, which results in the following formulation

$$\begin{cases} \min_{\boldsymbol{\beta}} \mathcal{L}_{ELM} = \frac{1}{2}||\boldsymbol{\beta}||^2 + C \cdot \frac{1}{2} \cdot \sum_{i=1}^{N} \boldsymbol{\xi}_i^2 \\ s.t.\ \mathrm{h}(\mathbf{x}_i)\boldsymbol{\beta} = \mathbf{t}_i - \boldsymbol{\xi}_i, i = 1, \ldots, N \end{cases} \tag{1}$$

where $\boldsymbol{\xi}_i$ denotes the prediction error w.r.t. the i-th training pattern, and C is a penalty constant on the training errors.

By substituting the constraint term in (1) into the objective function, an equivalent unconstrained optimization problem can be obtained as follows

$$\min_{\boldsymbol{\beta}\in\mathbb{R}^{L\times m}} \mathcal{L}_{ELM} = \frac{1}{2}||\boldsymbol{\beta}||^2 + C \cdot \frac{1}{2} \cdot ||\mathbf{T} - \mathbf{H}\boldsymbol{\beta}||^2 \tag{2}$$

where $\mathbf{H} = [h(\mathbf{x}_1); h(\mathbf{x}_2); \ldots; h(\mathbf{x}_N)] \in \mathbb{R}^{N\times L}$ and $\mathbf{T} = [\mathbf{t}_1, \mathbf{t}_2, \ldots, \mathbf{t}_N]^{\mathrm{T}}$.

The optimization problem (2) is a well known regularized least square problem. The closed form solution of $\boldsymbol{\beta}$ can be easily solved by setting the gradient of the subjective function (2) w.r.t. $\boldsymbol{\beta}$ to zero.

There are two cases when solving $\boldsymbol{\beta}$, i.e. if the number N of training patterns is larger than L, the gradient equation is over-determined, and the closed form solution can be obtained as

$$\boldsymbol{\beta}^* = \left(\mathbf{H}^{\mathrm{T}}\mathbf{H} + \frac{\mathbf{I}_L}{C}\right)^{-1}\mathbf{H}^{\mathrm{T}}\mathbf{T} \tag{3}$$

where $\mathbf{I}_L$ denotes the identity matrix with size of L.

If the number N of training patterns is smaller than L, an under-determined least square problem would be handled. In this case, the solution of (2) can be obtained as

$$\boldsymbol{\beta}^* = \mathbf{H}^{\mathrm{T}}\left(\mathbf{H}\mathbf{H}^{\mathrm{T}} + \frac{\mathbf{I}_N}{C}\right)^{-1}\mathbf{T} \tag{4}$$

where $\mathbf{I}_N$ denotes the identity matrix with size of N.

Therefore, in classifier training of ELM, the output weights can be computed by using (3) or (4) which depends on the number of training instances and the number of hidden nodes.

3 Proposed Domain Adaptation Transfer ELM

3.1 Source Domain Adaptation Transfer ELM (TELM-SDA)

Suppose that the source domain and target domain are represented D_S and D_T. In this paper, we assume that all the samples in the source domain are labeled data. The proposed *TELM-SDA* aims to learn a classifier β_S using a number of labeled instances from the source domain, and set the very few labeled data from the target domain as an appropriate regularizer for adapting to the source domain. The TELM-SDA can be formulated as

$$\min_{\beta_S,\xi_S^i,\xi_T^i} \frac{1}{2}\|\beta_S\|^2 + C_S\frac{1}{2}\sum_{i=1}^{N_S}\left(\xi_S^i\right)^2 + C_T\frac{1}{2}\sum_{j=1}^{N_T}\left(\xi_T^j\right)^2 \tag{5}$$

$$\text{s.t.}\begin{cases} H_S^i\beta_S = t_S^i - \xi_S^i, i = 1,\dots,N_S \\ H_T^j\beta_S = t_T^j - \xi_T^j, j = 1,\dots,N_T \end{cases}$$

where $H_S^i \in \mathbb{R}^{1\times L}, \xi_S^i \in \mathbb{R}^{1\times m}, t_S^i \in \mathbb{R}^{1\times m}$denote the output of hidden layer, the prediction error and the label with respect to the i-th training instance x_S^i from the source domain, $H_T^j \in \mathbb{R}^{1\times L}, \xi_T^j \in \mathbb{R}^{1\times m}, t_T^j \in \mathbb{R}^{1\times m}$ denote the output of hidden layer, the prediction error and the label with respect to the j-th guide samples x_T^j from the target domain, $\beta_S \in \mathbb{R}^{L\times m}$ is the output weights being solved, N_S and N_T denote the number of training instances and guide samples from the source domain and target domain, respectively, C_S and C_T are the penalty coefficients on the prediction errors of the labeled training data from source domain and target domain, respectively. Note that we call the very few labeled samples in target domain as "guide samples" in this paper.

From (5), we can find that the very few labeled guide samples from target domain can assist the learning of β_S and realize the knowledge transfer between source domain and target domain by introducing the third term as regularization with the second constraint, which makes the feature mapping of the guide samples from target domain approximate the labels with the output weights β_S learned by the training data from the source domain. The structure of the proposed TELM-SDA algorithm is illustrated in Fig.1.

To solve the optimization (5), the Largange multiplier equation is formulated as

$$L\left(\beta_S, \xi_S^i, \xi_T^j, \alpha_S, \alpha_T\right) = \frac{1}{2}\|\beta_S\|^2 + C_S\frac{1}{2}\sum_{i=1}^{N_S}\left(\xi_S^i\right)^2 + C_T\frac{1}{2}\sum_{j=1}^{N_T}\left(\xi_T^j\right)^2 \tag{6}$$
$$-\alpha_S\left(H_S^i\beta_S - t_S^i + \xi_S^i\right) - \alpha_T\left(H_T^i\beta_T - t_T^i + \xi_T^i\right)$$

By setting the partial derivation with respect to $\beta_S, \xi_S^i, \xi_T^j, \alpha_S, \alpha_T$ as zero, we have

$$\begin{cases} \frac{\partial L}{\partial \beta_S} = 0 \rightarrow \beta_S = H_S^{\mathrm{T}}\alpha_S + H_T^{\mathrm{T}}\alpha_T \\ \frac{\partial L}{\partial \xi_S} = 0 \quad \rightarrow \quad \alpha_S = C_S\xi_S \\ \frac{\partial L}{\partial \xi_T} = 0 \quad \rightarrow \quad \alpha_T = C_T\xi_T \\ \frac{\partial L}{\partial \alpha_S} = 0 \rightarrow H_S\beta_S - t_S + \xi_S = 0 \\ \frac{\partial L}{\partial \alpha_T} = 0 \rightarrow H_T\beta_S - t_T + \xi_T = 0 \end{cases} \tag{7}$$

where H_S and H_T are the output matrix of hidden layer with respect to the labeled data from source domain and target domain, respectively.

To solve β_S, α_S and α_T should be solve first. For the case that the number of training samples N_S is smaller than L ($N_S<L$), we substitute the 1st, 2nd, and 3rd equations into the 4th and 5th equations in (7), there is

$$\begin{cases} H_T H_S^{\mathrm{T}}\alpha_S + \left(H_T H_T^{\mathrm{T}} + \frac{I}{C_T}\right)\alpha_T = t_T \\ H_S H_T^{\mathrm{T}}\alpha_T + \left(H_S H_S^{\mathrm{T}} + \frac{I}{C_S}\right)\alpha_S = t_S \end{cases} \tag{8}$$

Let $H_T H_S^{\mathrm{T}} = A, H_T H_T^{\mathrm{T}} + \frac{I}{C_T} = B, H_S H_T^{\mathrm{T}} = C, H_S H_S^{\mathrm{T}} + \frac{I}{C_S} = D$, then eq.(8) can be written as

$$\begin{cases} A\alpha_S + B\alpha_T = t_T \\ C\alpha_T + D\alpha_S = t_S \end{cases} \rightarrow \begin{cases} B^{-1}A\alpha_S + \alpha_T = B^{-1}t_T \\ C\alpha_T + D\alpha_S = t_S \end{cases} \tag{9}$$

Then α_S and α_T can be solved as

$$\begin{cases} \alpha_S = (CB^{-1}A - D)^{-1}(CB^{-1}t_T - t_S) \\ \alpha_T = B^{-1}t_T - B^{-1}A(CB^{-1}A - D)^{-1}(CB^{-1}t_T - t_S) \end{cases} \tag{10}$$

According to the 1st equation in (7), we can obtain the output weights as

$$\begin{aligned}\beta_S &= H_S^{\mathrm{T}}\alpha_S + H_T^{\mathrm{T}}\alpha_T \\ &= H_S^{\mathrm{T}}(CB^{-1}A - D)^{-1}(CB^{-1}t_T - t_S) + \\ &\quad H_T^{\mathrm{T}}[B^{-1}t_T - B^{-1}A(CB^{-1}A - D)^{-1}(CB^{-1}t_T - t_S)]\end{aligned} \tag{11}$$

where $A = H_T H_S^{\mathrm{T}}, B = H_T H_T^{\mathrm{T}} + \frac{I}{C_T}, C = H_S H_T^{\mathrm{T}}, D = H_S H_S^{\mathrm{T}} + \frac{I}{C_S}$, I is the identity matrix with size of N_S.

For the case that the number of training samples N_S is larger than L ($N_S > L$), we can obtain from the 1st equation in (7) that $\alpha_S = (H_S H_S^{\mathrm{T}})^{-1}(H_S\beta_S - H_S H_T^{\mathrm{T}}\alpha_T)$, which is substituted into the 4th and 5th equations, then we calculate the output weights β_S as follows

$$\begin{aligned}&\begin{cases} H_S\beta_S + \xi_S = t_S \\ H_T\beta_S + \xi_T = t_T \end{cases} \rightarrow \begin{cases} H_S\beta_S + \frac{I}{C_S}\alpha_S = t_S \\ H_T\beta_S + \frac{I}{C_T}\alpha_T = t_T \end{cases} \rightarrow \begin{cases} H_S^{\mathrm{T}} H_S\beta_S + \frac{I}{C_S} H_S^{\mathrm{T}}\alpha_S = H_S^{\mathrm{T}} t_S \\ H_T\beta_S + \frac{I}{C_T}\alpha_T = t_T \end{cases} \\ &\rightarrow \begin{cases} H_S^{\mathrm{T}} H_S\beta_S + \frac{I}{C_S} H_S^{\mathrm{T}}(H_S H_S^{\mathrm{T}})^{-1}(H_S\beta_S - H_S H_T^{\mathrm{T}}\alpha_T) = H_S^{\mathrm{T}} t_S \\ \alpha_T = C_T(t_T - H_T\beta_S) \end{cases} \\ &\rightarrow \begin{cases} \left[H_S^{\mathrm{T}} H_S + \frac{I}{C_S} H_S^{\mathrm{T}}(H_S H_S^{\mathrm{T}})^{-1} H_S\right]\beta_S - \frac{I}{C_S} H_S^{\mathrm{T}}(H_S H_S^{\mathrm{T}})^{-1} H_S H_T^{\mathrm{T}}\alpha_T = H_S^{\mathrm{T}} t_S \\ \alpha_T = C_T(t_T - H_T\beta_S) \end{cases} \\ &\rightarrow \left(H_S^{\mathrm{T}} H_S + \frac{I}{C_S} + \frac{C_T}{C_S} H_T^{\mathrm{T}} H_T\right)\beta_S = H_S^{\mathrm{T}} t_S + \frac{C_T}{C_S} H_T^{\mathrm{T}} t_T \\ &\rightarrow \beta_S = (I + C_S H_S^{\mathrm{T}} H_S + C_T H_T^{\mathrm{T}} H_T)^{-1}(C_S H_S^{\mathrm{T}} t_S + C_T H_T^{\mathrm{T}} t_T)\end{aligned} \tag{12}$$

where I is the identity matrix with size of L.

In fact, the optimization (5) can be reformulated an equivalent unconstrained optimization problem in a matrix form by substituting the constraints into the objective function as

$$\min_{\beta_S} L_{TELM-SDA}(\beta_S) = \frac{1}{2}\|\beta_S\|^2 + C_S\frac{1}{2}\|t_S - H_S\beta_S\|^2 + C_T\frac{1}{2}\|t_T - H_T\beta_S\|^2 \tag{13}$$

By setting the gradient of $L_{TELM-SDA}$ with respect to β_S as zero, there is

$$\nabla L_{TELM-SDA} = \beta_S - C_S H_S^{\mathrm{T}}(t_S - H_S\beta_S) - C_T H_T^{\mathrm{T}}(t_T - H_T\beta_S) = 0 \tag{14}$$

Then, we can easily solve the β_S as formulated in (12).

For recognition of the numerous unlabeled data in target domain, we calculate the final output using the following

$$y_{Tu}^{k} = H_{Tu}^{k} \cdot \beta_S, k = 1, \dots, N_{Tu} \tag{15}$$

where H_{Tu}^{k} denote the hidden layer output with respect to the k-th unlabeled vector in target domain, and N_{Tu} is the number of unlabeled vectors in target domain.

In terms of the above discussion, the *TELM-SDA* algorithm is summarized as **Algorithm 1**.

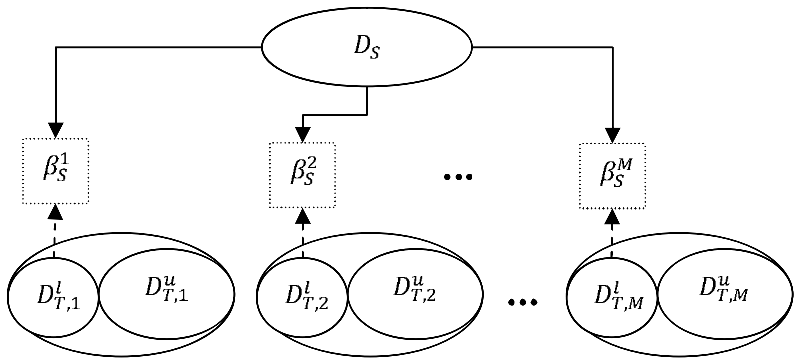

Fig. 1. Structure of TELM-SDA algorithm with M target domains (M tasks). The solid arrow denotes the training data from source domain D_S and the dashed arrow denotes the tiny labeled guide data from target domain D_T^l for classifier learning. The unlabeled data from D_T^u are not used.

Algorithm 1. TELM-SDA algorithm

```
Input:
  Training samples $\{X_S, t_S\} = \{x_S^i, t_S^i\}_{i=1}^{Ns}$ of the source domain $S$;
  Labeled guide samples $\{X_T, t_T\} = \{x_T^j, t_T^j\}_{j=1}^{N_T}$ of the target domain $T$;
  The tradeoff parameter $C_S$ and $C_T$ for source and target domain $T$.
Output:
  The output weights $\beta_S$;
  The predicted output $y_{Tu}$ of unlabeled data in target domain.
Procedure:
  1. Initialize the ELM network of $L$ hidden neurons with random input weights $W$ and hidden bias $B$.
  2. Calculate the output matrix $H_S$ and $H_T$ of hidden layer with source and target domains as $H_S = h(W \cdot X_S + B)$ and $H_T = h(W \cdot X_T + B)$.
  3. If $N_S<L$, compute the output weights $\beta_S$ using (11);
     Else, compute the output weights $\beta_S$ using (12).
  4. Calculate the predicted output $y_{Tu}$ using (15).
Return The output weights $\beta_S$ and predicted output $y_{Tu}$.
```

3.2 Target Domain Adaptation Transfer ELM (TELM-TDA)

In the proposed TELM-SDA, the classifier β_S is learned on the source domain with the very few labeled guide samples from the target domain as regularization. Study demonstrates that unlabeled data can also improve the performance of classification [17]. While the proposed TELM-TDA aims to learn a classifier β_T on the very few labeled guide samples from the target domain, and fully explore the numerous unlabeled data in the target domain with a base classifier β_S trained in source domain. As illustrated, the proposed TELM-SDA is formulated as

$$\min_{\beta_T} L_{TELM-TDA}(\beta_T) = \frac{1}{2}||\beta_T||^2 + C_T\frac{1}{2}||t_T - H_T\beta_T||^2 + C_{Tu}\frac{1}{2}||H_{Tu}\beta_S - H_{Tu}\beta_T||^2 \tag{16}$$

where β_T denotes the learned classifier, C_T, H_T, t_T are the same as that in TELM-SDA, C_{Tu}, H_{Tu} denote the regularization parameter and the output matrix of the hidden layer with respect to the unlabeled data X_{Tu} in target domain D_T, where $D_T = X_T \cup X_{Tu}$. The first term is to against the over-fitting, the second term is the least square loss function, and the third term is the regularization which means the domain adaptation between source domain and target domain. Note that β_S is a base classifier trained in source domain. In this paper, regularized ELM is used to train a base classifier β_S by solving

$$\min_{\beta_S} L_{TELM-TDA}(\beta_S) = \frac{1}{2}||\beta_S||^2 + C_S\frac{1}{2}||t_S - H_S\beta_S||^2 \tag{17}$$

where C_S, t_S, H_S denote the same meaning as that in TELM-SDA.

The structure of the proposed TELM-SDA is described in Fig.2, from which we can see that the unlabeled data in target domain have also been explored. To solve the optimization (16), by setting the gradient of $L_{TdaTELM}$ with respect to β_T as zero, we then have

$$\nabla L_{TELM-TDA} = \beta_T - C_T H_T^{\mathrm{T}}(t_T - H_T\beta_T) - C_{Tu}H_{Tu}^{\mathrm{T}}(H_{Tu}\beta_S - H_{Tu}\beta_T) = 0 \tag{18}$$

If the number of training samples $N_T > L$, then we can have from (18)

$$\beta_T = (I + C_T H_T^{\mathrm{T}} H_T + C_{Tu}H_{Tu}^{\mathrm{T}}H_{Tu})^{-1}(C_T H_T^{\mathrm{T}} t_T + C_{Tu}H_{Tu}^{\mathrm{T}}H_{Tu}\beta_S) \tag{19}$$

where I is the identity matrix with size of L.

If the number of training samples $N_T < L$, we would like to obtain β_S of the proposed TELM-TDA by a unified ELM framework. Let $t_{Tu} = H_{Tu}\beta_S$, the model (16) can be re-written as

$$\min_{\beta_T,\xi_T^i,\xi_{Tu}^i} \frac{1}{2}||\beta_T||^2 + C_T\frac{1}{2}\sum_{i=1}^{N_T}\left(\xi_T^i\right)^2 + C_{Tu}\frac{1}{2}\sum_{j=1}^{N_{Tu}}\left(\xi_{Tu}^j\right)^2 \tag{20}$$

$$\text{s.t.}\begin{cases} H_T^i\beta_T = t_T^i - \xi_T^i, i = 1, \dots, N_T \\ H_{Tu}^j\beta_T = t_{Tu}^j - \xi_{Tu}^j, j = 1, \dots, N_{Tu} \end{cases}$$

The Lagrange multiplier equation of (20) can be written as

$$L(\beta_T, \xi_T^i, \xi_{Tu}^i, \alpha_T, \alpha_{Tu}) = \frac{1}{2}\|\beta_T\|^2 + C_T\frac{1}{2}\sum_{i=1}^{N_T}(\xi_T^i)^2 + C_{Tu}\frac{1}{2}\sum_{j=1}^{N_{Tu}}(\xi_{Tu}^j)^2$$
$$-\alpha_T(H_T^i\beta_T - t_T^i + \xi_T^i) - \alpha_{Tu}(H_{Tu}^i\beta_T - t_{Tu}^i + \xi_{Tu}^i) \quad (21)$$

By setting the partial derivation with respect to $\beta_T, \xi_T^i, \xi_{Tu}^j, \alpha_T, \alpha_{Tu}$ as zero, we have

$$\begin{cases} \frac{\partial L}{\partial \beta_T} = 0 \rightarrow \beta_T = H_T^{\mathrm{T}}\alpha_T + H_{Tu}^{\mathrm{T}}\alpha_{Tu} \\ \frac{\partial L}{\partial \xi_T} = 0 \rightarrow \alpha_T = C_T\xi_T \\ \frac{\partial L}{\partial \xi_{Tu}} = 0 \rightarrow \alpha_{Tu} = C_{Tu}\xi_{Tu} \\ \frac{\partial L}{\partial \alpha_T} = 0 \rightarrow H_T\beta_T - t_T + \xi_T = 0 \\ \frac{\partial L}{\partial \alpha_{Tu}} = 0 \rightarrow H_{Tu}\beta_T - t_{Tu} + \xi_{Tu} = 0 \end{cases} \quad (22)$$

To solve β_T, let $H_{Tu}H_T^{\mathrm{T}} = O, H_{Tu}H_{Tu}^{\mathrm{T}} + \frac{I}{C_{Tu}} = P, H_T H_{Tu}^{\mathrm{T}} = Q, H_T H_T^{\mathrm{T}} + \frac{I}{C_T} = R$, with similar calculation of (8), (9), and (10),we can get

$$\begin{cases} \alpha_T = (QP^{-1}O - R)^{-1}(QP^{-1}t_{Tu} - t_T) \\ \alpha_{Tu} = P^{-1}t_{Tu} - P^{-1}O(QP^{-1}O - R)^{-1}(QP^{-1}t_{Tu} - t_T) \end{cases} \quad (23)$$

Therefore, the output weights if $N_T<L$ can be obtained as

$$\begin{aligned} \beta_T &= H_T^{\mathrm{T}}\alpha_T + H_{Tu}^{\mathrm{T}}\alpha_{Tu} \\ &= H_T^{\mathrm{T}}(QP^{-1}O - R)^{-1}(QP^{-1}t_{Tu} - t_T) \\ &\quad + H_{Tu}^{\mathrm{T}}[P^{-1}t_{Tu} - P^{-1}O(QP^{-1}O - R)^{-1}(QP^{-1}t_{Tu} - t_T)] \end{aligned} \quad (24)$$

where $t_{Tu} = H_{Tu}\beta_S, O = H_{Tu}H_T^{\mathrm{T}}, P = H_{Tu}H_{Tu}^{\mathrm{T}} + \frac{I}{C_{Tu}}, Q = H_T H_{Tu}^{\mathrm{T}}, R = H_T H_T^{\mathrm{T}} + \frac{I}{C_T}$, I is the identity matrix with size of N_T.

For recognition of the numerous unlabeled data in target domain, we calculate the final output using the following

$$y_{Tu}^k = H_{Tu}^k \cdot \beta_T, k = 1, \ldots, N_{Tu} \quad (25)$$

where H_{Tu}^k denote the hidden layer output with respect to the k-th unlabeled vector in target domain, and N_{Tu} is the number of unlabeled vectors in target domain.

In terms of the above discussion, the *TELM-TDA* algorithm is summarized as **Algorithm 2**.

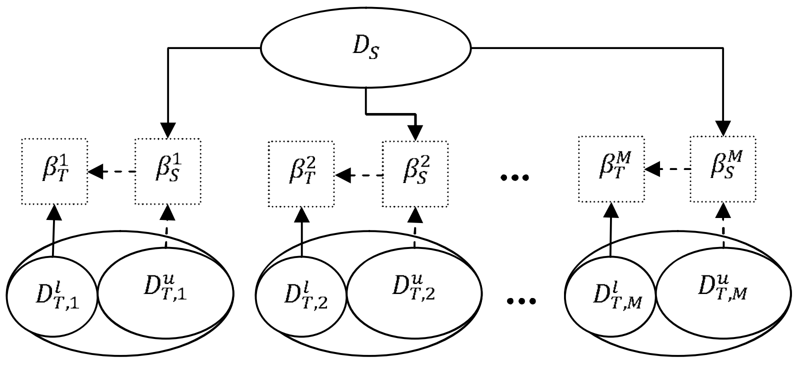

Fig. 2. Structure of TELM-TDA algorithm with M target domains (M tasks). The solid arrow connected with D_S denotes the training for base classifier β_S, the dashed line connected with D_T^u denotes the tentative test of base classifier using the unlabeled data from target domain, the solid arrow connected with D_T^l denotes the terminal classifier learning of β_T, and the dashed arrow connected between β_S and β_T denotes the regularization for learning β_T.

```
Algorithm 2. TELM-TDA algorithm
Input:
    Training samples $\{X_S, t_S\} = \{x_S^i, t_S^i\}_{i=1}^{N_S}$ of the source domain $S$;
    Labeled guide samples $\{X_T, t_T\} = \{x_T^j, t_T^j\}_{j=1}^{N_T}$ of the target domain $T$;
    Unlabeled samples $\{X_{Tu}\} = \{x_{Tu}^k\}_{k=1}^{N_{Tu}}$ of the target domain $T$;
    The tradeoff parameters $C_S$, $C_T$ and $C_{Tu}$.
Output:
    The output weights $\beta_T$;
    The predicted output $y_{Tu}$ of unlabeled data in target domain.
Procedure:
    1. Initialize the ELM network of $L$ hidden neurons with random input weights $W_1$ and hidden bias $B_1$.
    2. Calculate the output matrix $H_S$ of hidden layer with source domain as $H_S = h(W_1 \cdot X_S + B_1)$.
    3. If $N_S<L$, compute the output weights $\beta_S$ of the base classifier using (4);
       Else, compute the output weights $\beta_S$ of the base classifier using (3).
    4. Initialize the ELM network of $L$ hidden neurons with random input weights $W_2$ and hidden bias $B_2$.
    5. Calculate the output matrix $H_T$ and $H_{Tu}$ of hidden layer with labeled and unlabeled data in target domains as $H_T = h(W_2 \cdot X_T + B_2)$ and $H_{Tu} = h(W_2 \cdot X_{Tu} + B_2)$.
    6. If $N_T<L$, compute the output weights $\beta_T$ using (24);
       Else, compute the output weights $\beta_T$ using (19).
    7. Calculate the predicted output $y_{Tu}$ using (25).
Return The output weights $\beta_T$ and predicted output $y_{Tu}$.
```

Remark 1: From the proposed source domain adaptation transfer ELM (TELM-SDA) and target domain adaptation transfer ELM (TELM-TDA), we can observe that two stages are included namely feature mapping with random selected weights and biases and output weights learning which are the main parts in ELM. For ELM, the only information in source domain is considered. However, for domain adaptation transfer ELM, very few labeled samples from target domain are explored without changing the unified ELM framework. The common differences of the ELMs lie in the calculation of output weights. The unified framework for TELM-SDA and TELM-TDA might draw some new perspectives for developing the ELMs.

Remark 2: We can observe that the TELM-SDA and TELM-TDA have similar structure in model and algorithm, except for the base classifier learning in TELM-TDA. However, the essential difference lies in that the numerous unlabeled data which may be useful for improving generalization performance are also explored in TELM-TDA. Specifically, TELM-SDA trains a classifier using the information of source domain but draw some knowledge with labeled guide samples from the target domain. In this way, the knowledge from target domain can be effectively transferred to source domain through appropriate models. Instead, TELM-TDA aims to train a classifier using the guide data from target domain but introduce a regularizer through exploring the unlabeled data and a base classifier trained from source domain.

4 Experiments

In this section, we will employ the proposed TELM-SDA and TELM-TDA algorithms on olfactory data collected by electronic nose for sensor drift compensation. Electronic nose is an artificial olfaction system, which is developed for gas recognition [18, 19], tea quality assessment [20, 21], medical diagnosis [22], environmental monitor and gas concentration estimation [23, 24], etc. by using pattern recognition and gas sensor array with cross-sensitivity and broad spectrum characteristics. However, gas sensor drift will be caused due to the change of internal component and aging, which would reduce the generalization performance of well trained classifier [25]. Therefore, researchers have to retrain the classifier using a number of new samples in a period regularly. The tedious work for classifier retraining and acquisition of new labeled samples regularly seems to be impossible, due to the complicated experiments of electronic nose. Though researchers have paid more attention to sensor drift and aim to find some effective ways for drift compensation through classifier ensembles and drift prediction [16, 26-29], sensor drift is still a challenging issue in machine olfaction community and sensory field. To our best knowledge, there are no very effective methods for dealing with sensor drift. Therefore, we aim to enhance the adaptive performance of classifiers to drifted data with very low complexity and little work. It would be very meaningful and interesting to train a classifier using very few labeled new samples (target domain) as guide samples without giving up the recognized “useless” old data (source domain), and make the new trained classifier adapt to the new patterns in target domain.

4.1 Experimental Data

For verification of the proposed TELM-SDA and TELM-TDA algorithms, the long-term sensor drift big data of three years which was released in UCI Machine Learning Repository [31] by Vergara *et al.* [26, 30] has been explored in this paper.

This dataset contains 13,910 measurements (observation samples) from an electronic nose system with 16 gas sensors exposed to 6 kinds of pure gaseous substances including acetone, acetaldehyde, ethanol, ethylene, ammonia, and toluene at different concentration levels. The sensor drift big dataset was gathered during the period of January 2008 to February 2011 with 36 months in a gas delivery platform. For each sensor, 8 features were extracted, and results in a 128-dimensional feature vector (8 features × 16 sensors) for each measurement. We refer readers to [26] for specific technical details on how to select the 8 features for single sensor. Totally, 10 batches of data are included in the dataset which was divided according to months. The details of the dataset have been presented in Table 1.

Table 1. Number of samples for each subject in the sensor drifted big data

Batch ID	Month	Acetone	Acetaldehyde	Ethanol	Ethylene	Ammonia	Toluene	Total
Batch 1	1, 2	90	98	83	30	70	74	445
Batch 2	3-10	164	334	100	109	532	5	1244
Batch 3	11-13	365	490	216	240	275	0	1586
Batch 4	14, 15	64	43	12	30	12	0	161
Batch 5	16	28	40	20	46	63	0	197
Batch 6	17-20	514	574	110	29	606	467	2300
Batch 7	21	649	662	360	744	630	568	3613
Batch 8	22, 23	30	30	40	33	143	18	294
Batch 9	24, 30	61	55	100	75	78	101	470
Batch 10	36	600	600	600	600	600	600	3600

4.2 Experimental Setup

We follow the experimental setup in [26] to evaluate the proposed domain adaptation transfer ELM models. The number of hidden neurons L is set as 1000. The features are scaled appropriately to lie between -1 and +1. The RBF function is used as the activation function in the hidden layer (i.e. feature mapping function) in which the kernel width is set as 1. In TELM-SDA model, the penalty coefficients C_S and C_T are set as 0.01 and 10 throughout the experiments, respectively. In TELM-TDA model, the penalty coefficient C_S for base classifier is set as 0.001, C_T and C_{Tu} are set as 0.001 and 100 throughout the experiments, respectively. For effective verification of the proposed methods, two experimental settings according to [16] are given as follows:

Setting 1: Take batch 1 (source domain) as fixed training set and tested on other 9 batches (target domains);

Setting 2: The training set (source domain) is dynamically changed with batch K-1 and tested on batch K (target domain), K=2,…,10.

For studying the relation between the number *k* of labeled samples in target domain and the recognition accuracy, *k* is tried in the set of {5, 10, 15, 20, 25, 30, 35, 40, 45, 50}. In addition, for comparisons, we have compared with multi-class SVM with RBF kernel (SVM-rbf), the geodesic flow kernel (SVM-gfk), and the combination kernel (SVM-comgfk). Besides, we also compared with the semi-supervised methods such as manifold regularization with RBF kernel (ML-rbf) and manifold regularization with combination kernel (ML-comgfk), which have been presented in [16, 26] for the same sensor drift data. Additionally, the regularized ELM with RBF function in hidden layer (ELM-rbf) from [29] is also compared in our experiments. In experiments, we run the ELM-rbf, TELM-SDA and TELM-TDA 10 times, and the average value for each item is provided.

4.3 Results

Under the consideration above, we employ the experiments on **Setting 1** and **Setting 2**, respectively. The comparisons under setting 1 with recognition accuracy of 9 batches for different methods are presented in Table 2. We have shown two conditions of TELM-SDA with 20 labeled guide samples and 30 labeled guide samples. For TELM-TDA, 40 and 50 labeled samples from the target domain are used, respectively, considering that TELM-TDA trains a classifier using the labeled samples from the target domain, therefore, more labeled samples would be necessary which is slightly different from TELM-SDA. From Table 2, it can be obviously seen that the proposed TELM-SDA and TELM-TDA are much better than other existing methods including SVM with different kernels, manifold regularization with different kernels. For TELM-SDA and TELM-TDA, the testing accuracies on batch 2-10 with a training classifier using the data in batch 1 can still be feasible without performance reduction. This means that the sensor drift can be compensated very well with domain adaptation knowledge transfer. For visually observing the change of performance with sensor drift, we show the recognition accuracy on batches successively as Fig. 3. Through the results of the regularized ELM, we can see that the generalization performance and knowledge transfer capability have been well improved by the proposed TELM-SDA and TELM-TDA with domain adaptation. Comparison between TELM-SDA and TELM-TDA, the latter needs more labeled samples than the former. From the computational complexity, due to that there is a base classifier in TELM-TDA, TELM-SDA would be more appropriate in real-world applications which considers the data in source domain and very few labeled guide samples from target domain for classifier learning. The specific comparisons between TELM-SDA and TELM-TDA will be employed later.

Table 2. Comparisons of recognition accuracy (%) under the experimental **Setting 1**, i.e. trained on batch 1 and tested on other successive 9 batches

Batch ID	Batch 2	Batch 3	Batch 4	Batch 5	Batch 6	Batch 7	Batch 8	Batch 9	Batch 10
SVM-rbf	74.36	61.03	50.93	18.27	28.26	28.81	20.07	34.26	34.47
SVM-gfk	72.75	70.08	60.75	75.08	73.82	54.53	55.44	69.62	41.78
SVM-comgfk	74.47	70.15	59.78	75.09	73.99	54.59	55.88	70.23	41.85
ML-rbf	42.25	73.69	75.53	66.75	77.51	54.43	33.50	23.57	34.92
ML-comgfk	80.25	74.99	78.79	67.41	77.82	71.68	49.96	50.79	53.79
ELM-rbf	70.63	66.44	66.83	63.45	69.73	51.23	49.76	49.83	33.50
TELM-SDA(20)	87.57	96.53	82.61	81.47	84.97	71.89	78.10	87.02	57.42
TELM-SDA(30)	87.98	95.74	85.16	95.99	94.14	83.51	86.90	100.0	53.62
TELM-TDA(40)	83.52	96.34	88.20	99.49	78.43	80.93	87.42	100.0	56.25
TELM-TDA(50)	97.96	95.34	99.32	99.24	97.03	83.09	95.27	100.0	59.45

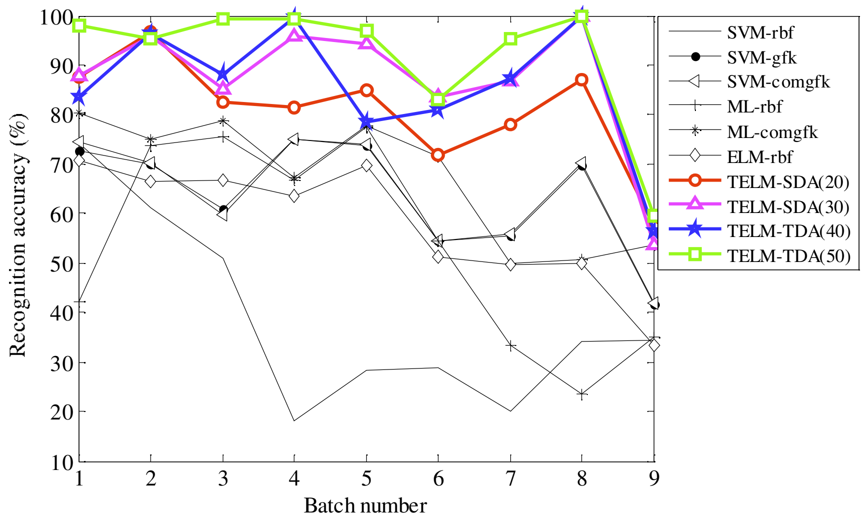

Fig. 3. Comparisons of different methods in **Setting 1**

From the experimental results in **Setting 1**, the proposed methods perform better results, and the sensor drift can be well compensated. We have also employed the experiments by following Setting 2 i.e. trained on batch *K*-1 and tested on batch *K*, for which the results are presented in Table 3. We can find that the proposed domain adaptation transfer ELM performs much better than other baseline methods for sensor drift big data. The visual insight of these methods in setting 2 has been described in Fig.4 which shows the robust performance of the proposed methods in sensor drift compensation and knowledge transfer.

Table 3. Comparisons of recognition accuracy (%) under the experimental **Setting 2**. i.e. trained on batch *K*-1, and tested on batch *K* ($2 \leq K \leq 10$).

Batch ID	1→2	2→3	3→4	4→5	5→6	6→7	7→8	8→9	9→10
SVM-rbf	74.36	87.83	90.06	56.35	42.52	83.53	91.84	62.98	22.64
SVM-gfk	72.75	74.02	77.83	63.91	70.31	77.59	78.57	86.23	15.76
SVM-comgfk	74.47	73.75	78.51	64.26	69.97	77.69	82.69	85.53	17.76
ML-rbf	42.25	58.51	75.78	29.10	53.22	69.17	55.10	37.94	12.44
ML-comgfk	80.25	98.55	84.89	89.85	75.53	91.17	61.22	95.53	39.56
ELM-rbf	70.63	40.44	64.16	64.37	72.70	80.75	88.20	67.00	22.00
TELM-SDA(20)	87.57	96.90	85.59	95.89	80.53	91.56	88.71	88.40	45.61
TELM-SDA(30)	87.98	96.58	89.75	99.04	84.43	91.75	89.83	100.0	58.44
TELM-TDA(40)	83.52	96.41	81.36	96.45	85.13	80.49	85.71	100.0	56.81
TELM-TDA(50)	97.96	95.62	99.63	98.17	97.13	83.10	94.90	100.0	59.88

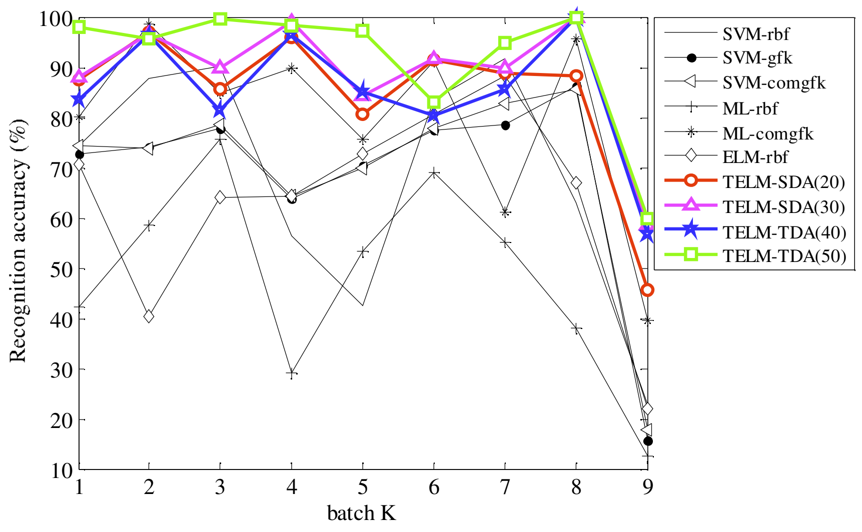

Fig. 4. Comparisons of different methods in **Setting 2**

5 Conclusion

In this paper, two ELM based algorithms, TELM-SDA and TELM-TDA have been proposed to extend the ELMs for learning tasks with multi-domains, respectively. Through the sensor drift big data analysis in machine olfaction, the proposed domain adaptation transfer ELMs consistently outperform the existing methods such as SVMs, semi-supervised manifold regularizations and ELMs for sensor drift compensation. For dealing with large scale sensor drift data collected by an electronic nose, the proposed methods have also the advantages of ELMs including high efficiency of classifier/predictor learning and straightforward implementation of multi-class classification. The adaptation of multi-domain sensor drift big data can be efficiently and effectively implemented by using the proposed domain adaptation transfer ELMs. More importantly, the proposed methods can also provide new perspectives for exploring ELM theory. Experimental results demonstrate that the proposed methods can obvious improve the transfer capability of ELMs in real-world applications.

Acknowledgement. This work was supported by National Natural Science Foundation of China (61401048), Hong Kong Scholar Program (No.XJ2013044) and also funded by China Postdoctoral Science Foundation (No. 2014M550457).

References

1. Huang, G.B., Zhu, Q.Y., Siew, C.K.: Extreme learning machine: Theory and applications. Neurocomputing 70, 489–501 (2006)
2. Feng, G., Huang, G.B., Lin, Q., Gay, R.: Error minimized extreme learning machine with growth of hidden nodes and incremental learning. IEEE Trans. Neural Netw. 20, 1352–1357 (2009)
3. Huang, G.B., Zhou, H., Ding, X., Zhang, R.: Extreme Learning Machine for Regression and Multiclass Classification. IEEE Trans. Systems, Man, Cybernetics: Part B 42, 513–529 (2012)
4. Huang, G.B., Chen, L., Siew, C.K.: Universal approximation using incremental constructive feedforward networks with random hidden nodes. IEEE Trans. Neural. Netw. 17, 879–892 (2006)
5. Zhu, Q.Y., Qin, A.K., Suganthan, P.N., Huang, G.B.: Evolutionary extreme learning machine. Pattern Recognition 38, 1759–1763 (2005)
6. Zong, W., Huang, G.B., Chen, Y.: Weighted extreme learning machine for imbalance learning. Neurocomputing 101, 229–242 (2013)
7. Bai, Z., Huang, G.B., Wang, D., Wang, H., Brandon Westover, M.: Sparse Extreme Learning Machine for Classification. IEEE Trans. Cybernetics (2014)
8. Li, X., Mao, W., Jiang, W.: Fast sparse approximation of extreme learning machine. Neurocomputing 128, 96–103 (2014)
9. Huang, G., Song, S., Gupta, J.N.D., Wu, C.: Semi-Supervised and Unsupervised Extreme Learning Machines. IEEE Trans. Cybernetics (2014)
10. Blitzer, J., McDonald, R., Pereira, F.: Domain adaptation with structural correspondence learning. In: Proc. Conf. Emp. Methods Natural Lang. Process, pp. 120–128 (2006)
11. Yang, J., Yan, R., Hauptmann, A.G.: Cross-domain video concept detection using adaptive SVMs. In: Proc. Int. Conf. Multimedia, pp. 188–197 (2007)
12. Pan, S.J., Tsang, I.W., Kwok, J.T., Yang, Q.: Domain adaptation via transfer component analysis. IEEE Trans. Neural Netw. 22, 199–210 (2011)
13. Duan, L., Tsang, I.W., Xu, D., Chua, T.S.: Domain adaptation from multiple sources via auxiliary classifiers. Proc. Int. Conf. Mach. Learn., 289–296 (2009)
14. Duan, L., Xu, D., Tsang, I.W.: Domain Adaptation from Multiple Sources: Domain-Dependent Regularization Approach. IEEE Trans. Neur. Netw. Learn. Syst. 23, 504–518 (2012)
15. Gopalan, R., Li, R., Chellappa, R.: Domain adaptation for object recognition: An unsupervised approach. In: Proc. ICCV, pp. 999–1006 (2011)

16. Liu, Q., Li, X., Ye, M., Sam Ge, S., Du, X.: Drift Compensation for Electronic Nose by Semi-Supervised Domain Adaptation. IEEE Sensors Journal 14, 657–665 (2014)
17. Belkin, M., Niyogi, P., Sindhwani, V.: Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. J. Mach. Learn. Res. 7, 2399–2434 (2006)
18. Zhang, L., Tian, F.C.: A new kernel discriminant analysis framework for electronic nose recognition. Analytica Chimica Acta 816, 8–17 (2014)
19. Zhang, L., Tian, F., Nie, H., Dang, L., Li, G., Ye, Q., Kadri, C.: Classification of multiple indoor air contaminants by an electronic nose and a hybrid support vector machine. Sens. Actu. B. 174, 114–125 (2012)
20. Brudzewski, K., Osowski, S., Dwulit, A.: Recognition of coffee using differential electronic nose. IEEE Trans. Instru. Meas. 61, 1803–1810 (2012)
21. Tudu, B., Metla, A., Das, B., Bhattacharyya, N., Jana, A., Ghosh, D., Bandyopadhyay, R.: Towards Versatile Electronic Nose Pattern Classifier for Black Tea Quality Evaluation: An Incremental Fuzzy Approach. IEEE Trans. Instru. Meas. 58, 3069–3078 (2009)
22. Gardner, J.W., Shin, H.W., Hines, E.L.: An electronic nose system to diagnose illness. Sens. Actu. B. 70, 19–24 (2000)
23. Zhang, L., Tian, F., Kadri, C., Pei, G., Li, H., Pan, L.: Gases concentration estimation using heuristics and bio-inspired optimization models for experimental chemical electronic nose. Sens. Actu. B. 160, 760–770 (2011)
24. Zhang, L., Tian, F.: Performance Study of Multilayer Perceptrons in a Low-Cost Electronic Nose. IEEE Trans. Instru. Meas. 63 (2014)
25. Di Carlo, S., Falasconi, M.: Drift Correction Methods for Gas Chemical Sensors in Artificial Olfaction Systems: Techniques and Challenges. Advances in Chemical Sensors, pp. 305–326 (2012)
26. Vergara, A., Vembu, S., Ayhan, T., Ryan, M.A., Homer, M.L., Huerta, R.: Chemical gas sensor drift compensation using classifier ensembles. Sens. Actu. B. 167, 320–329 (2012)
27. Romain, A.C., Nicolas, J.: Long term stability of metal oxide-based gas sensors for e-nose environmental applications: An overview. Sens. Actu. B. 146, 502–506 (2010)
28. Zhang, L., Tian, F., Liu, S., Dang, L., Peng, X., Yin, X.: Chaotic time series prediction of E-nose sensor drift in embedded phase space. Sens. Actu. B. 182, 71–79 (2013)
29. Arul Pon Daniel, D., Thangavel, K., Manavalan, R., Chandra, S., Boss, R.: ELM-Based Ensemble Classifier for Gas Sensor Array Drift Dataset. Computational Intelligence, Cyber Security and Computational Models. Advances in Intelligent Systems and Computing 246, 89–96 (2014)
30. Lujan, I.R., Fonollosa, J., Vergara, A., Homer, M., Huerta, R.: On the calibration of sensor arrays for pattern recognition using the minimal number of experiments. Chemometrics and Intelligent Laboratory Systems 130, 123–134 (2014)
31. `http://archive.ics.uci.edu/ml/datasets/Gas+Sensor+Array+Drift+Dataset+at+Different+Concentrations`