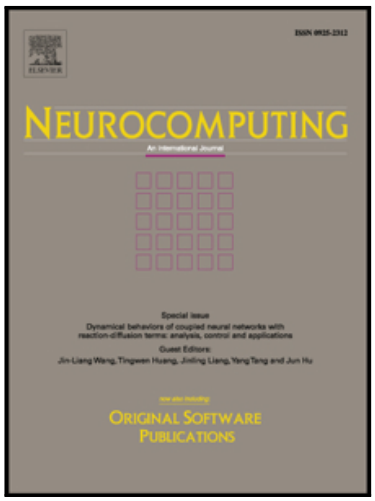

Deep Object Recognition Across Domains based on Adaptive Extreme Learning Machine

Lei Zhang , Zhenwei He , Yan Liu

PII: S0925-2312(17)30286-2
DOI: 10.1016/j.neucom.2017.02.016
Reference: NEUCOM 18082

To appear in: *Neurocomputing*

Received date: 26 October 2016
Revised date: 30 January 2017
Accepted date: 5 February 2017

Please cite this article as: Lei Zhang , Zhenwei He , Yan Liu , Deep Object Recognition Across Domains based on Adaptive Extreme Learning Machine, *Neurocomputing* (2017), doi: 10.1016/j.neucom.2017.02.016

Deep Object Recognition Across Domains based on Adaptive Extreme Learning Machine

Lei Zhang*, Zhenwei He, Yan Liu

College of Communication Engineering, Chongqing University, 174 ShaZheng street, ShaPingBa District, Chongqing, 400044, China

*Author to whom correspondence should be addressed; E-mail: leizhang@cqu.edu.cn

Abstract: Deep learning with a convolutional neural network (CNN) has been proved to be very effective in feature extraction and representation of images. For image classification problems, this work aims at exploring the capability of extreme learning machine on high-level deep features of images. Additionally, motivated by the biological learning mechanism of ELM, in this paper, an adaptive extreme learning machine (AELM) method is proposed for handling cross-task (domain) learning problems, without loss of its nature of randomization and high efficiency. The proposed AELM is an extension of ELM from single task to cross task learning, by introducing a new error term and Laplacian graph based manifold regularization term in objective function. We have discussed the nearest neighbor, support vector machines and extreme learning machines for image classification under deep convolutional activation feature representation. Specifically, we adopt 4 benchmark object recognition datasets from multiple sources with domain bias for evaluating different classifiers. The deep features of the object dataset are obtained by a well-trained CNN with five convolutional layers and three fully-connected layers on ImageNet. Experiments demonstrate that the proposed AELM is comparable and effective in single and multiple domains based recognition tasks.

Keywords: Deep learning; image classification; support vector machine; extreme learning machine; object recognition

1. Introduction

Recently, deep learning as the hottest learning technique has been widely explored in machine learning, computer vision, natural language processing and data mining. In the early, convolutional neural network (CNN), as the most important deep net in deep learning, has been applied to document recognition and face

recognition [1, 2]. Moreover, some deep learning algorithms with multi-layer fully connected networks (e.g. multi-layer perceptrons, MLP) for auto-encoder have been proposed, such as stacked auto encoders (SAE) [3], deep belief networks (DBN) [4] and deep Boltzmann machines (DBM) [5]. However, in large-scale learning problems, such as image classification in computer vision, CNNs with convolutioanl layers, pooling layers and fully-connected layers are widely investigated for its strong deep feature representation ability and state-of-the-art performance in challenged big datasets like ImageNet, Pascal VOC, etc. In the latest progress of deep learning, researchers have achieved new records in face verification by using different CNN structures [6, 7, 8, 9, 33]. The latest verification accuracy on LFW data is 99.7% by Face++ team. Besides the face recognition, CNN has also achieved very competitive results on ImageNet for image classification and Pascal VOC data [10-17]. From these works, CNNs have been proved to be highly effective for deep feature representation with large-scale parameters. The main advantages of deep learning can be shown in three facets. 1) Feature representation. CNN integrates feature extraction (raw pixels) and model learning together, without using any other advanced low-level feature descriptors. 2) Large-scale learning. With the adjustable network structures, big data in millions can be learned by a CNN at one time. 3) Parameter learning. Due to the scalable network structures, millions of parameters can be automatically trained. Therefore, CNN based deep methods can be state-of-the-art parameter learning techniques.

In this paper, we focus on the discussion on the superiority of the traditional classifiers such as ELMs in cross-domain image classification based on very high-level CNN deep feature representation. Briefly, the concept “domain” is often identified as a task with different distribution from source data and it can be understood as set or space. Cross-domain recognition means that the source domain and target domain generate similar high-level semantics but different distribution (i.i.d). Specifically, the nearest neighbor (NN) [18], support vector machine (SVM) [19], least-square support vector machine (LSSVM) [20], extreme learning machine (ELM) [21] and kernel extreme learning machine (KELM) [22] are studied. These classifiers are well-known in many different applications. Furthermore, for effectively handling cross-task learning and recognition problems presented in this paper, an adaptive extreme learning machine (AELM) is proposed. The “adaptive” concept is induced due to its classifier adaptation ability across tasks. ELM was initially proposed for generalized single-hidden-layer feed-forward neural networks and overcome the disadvantages such as local minima, learning rate, stopping criteria and learning epochs that exist in

gradient-based back-propagation (BP) algorithm. In recent years, ELMs are widely used because of some significant advantages such as learning speed, ease of implementation and minimal human intervention. Its high capability in large scale learning and artificial intelligence is witnessed. The main steps of ELM include random projection of hidden layer based on random input weights and analytically determined solution based on Moore-Penrose generalized inverse. ELM has been proved to be efficient and effective for regression and classification tasks [23, 24]. The latest work about the principles and brain-alike learning of ELM has been presented [25]. Many improvements and new applications of ELMs have been proposed by world-wide researchers. The newest work about improved extreme learning machines in deep auto-encoder, local receptive fields for deep learning, transfer learning, and semi-supervised learning have also been proposed [26-30, 36-43]. Yang *et al.* proposed a subnetwork nodes based multilayer ELM framework for representational learning [44]. The same author also proposed an autoencoder method for dimension reduction and reconstruction [45]. With the Mercer condition applied, a kernel ELM (KELM) that computes a kernel matrix of hidden layers has also been proposed [22]. Liu *et al.* proposed an extreme kernel sparse learning method for tactile object recognition [46]. A salient feature of KELM is that the random input weights and bias can be avoided. A sequential partial optimization algorithm was proposed and may be an interesting solver of ELM [47].

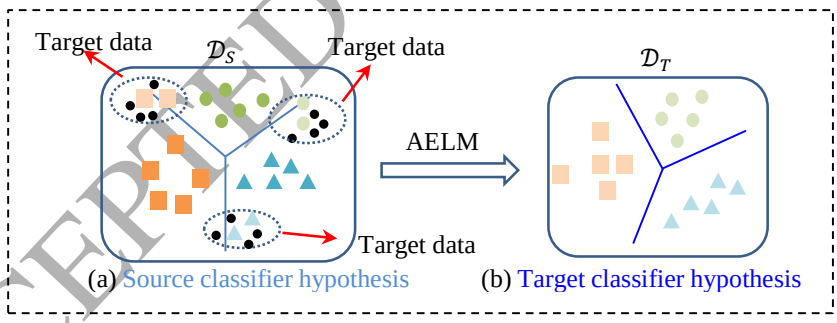

Fig. 1. Schematic diagram of AELM method. In (a), the circle points with black color denote unlabeled target data, the points with light color denote few labeled target data, and other points denote labeled source data of 3 classes. In (b), the learned target classifier hypothesis is obtained.

In this paper, we present a survey of NN, SVM, LSSVM, ELM and KELM for object recognition on the deep convolutional activation features trained by CNN on ImageNet, and have an insight of which one is the best for classification based on very high level deep representation. Furthermore, with the motivations of

ELM nature that 1) biological learning nature, 2) simplicity in structure, 3) fast classifier learning ability, and 4) single task learning flaw, an adaptive extreme learning machine (AELM) approach is specially proposed for addressing cross-task recognition problems without loss of generality. A detailed schematic of the proposed AELM method is described in Fig. 1. The proposed AELM is an extension of traditional ELM from single task to cross task learning, yet still inherits the advantage of ELM in its single hidden layer structure and high computational efficiency.

The rest of this paper is organized as follows. Section 2 presents a method review of support vector machines and extreme learning machines. The proposed adaptive extreme learning machine approach is formulated in Section 3. Section 4 shows the training and testing protocol of CNN for deep representation of images. Section 5 presents the experiments and results. The statistical significance test is implemented in Section 6. Finally, Section 7 concludes this paper.

2. Overview of SVMs And ELMs

2.1. Support Vector Machine (SVM)

In this section, the principle of SVM for classification problems is briefly reviewed. More details can be referred to [19]. Given a training set of N data points $\{\mathbf{x}_i, y_i\}_{i=1}^{N}$, where the label $y_i \in \{-1, 1\}, i = 1, \cdots, N$. According to the structural risk minimization principle, SVM aims at solving the following risk bound minimization problem with inequality constraint.

$$\begin{aligned} &\min_{\mathbf{w},\xi_i} \frac{1}{2}\|\mathbf{w}\|^2 + C \cdot \sum_{i=1}^{N} \xi_i, \\ &s.t.\ \xi_i \geq 0,\ y_i\left[\mathbf{w}^{\mathrm{T}}\varphi(\mathbf{x}_i) + b\right] \geq 1 - \xi_i \end{aligned} \tag{1}$$

where $\varphi(\cdot)$ is a linear/nonlinear mapping function, $\mathbf{w}$ and b are the parameters of classifier hyper-plane.

Generally, SVM can be transformed into its dual formulation with equality constraint by using Lagrange multiplier method. The Lagrange function is formulated as

$$\begin{aligned} L(\mathbf{w}, b, \xi_i; \alpha_i, \lambda_i) &= \frac{1}{2}\|\mathbf{w}\|^2 + C \cdot \sum_{i=1}^{N} \xi_i \\ &- \sum_{i=1}^{N} \alpha_i \left(y_i\left[\mathbf{w}^{\mathrm{T}}\varphi(\mathbf{x}_i) + b\right] - 1 + \xi_i\right) - \sum_{i=1}^{N} \lambda_i \xi_i \end{aligned} \tag{2}$$

where $\alpha_i \geq 0$ and $\lambda_i \geq 0$ are Lagrange multipliers. The solution is given as the saddle point of Lagrange function (2) by solving the following problem

$$\max_{\alpha_i,\lambda_i}\min_{\mathbf{w},b,\xi_i} L(\mathbf{w},b,\xi_i;\alpha_i,\lambda_i) \tag{3}$$

By calculating the partial derivatives of Eq. (2) with respect to **w**, b and ξ_i, one can obtain

$$\begin{cases} \dfrac{\partial L(\mathbf{w},b,\xi_i;\alpha_i,\lambda_i)}{\partial \mathbf{w}} = 0 \;\rightarrow\; \mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \varphi(\mathbf{x}_i) \\ \dfrac{\partial L(\mathbf{w},b,\xi_i;\alpha_i,\lambda_i)}{\partial b} = 0 \;\rightarrow\; \sum_{i=1}^{N} \alpha_i y_i = 0 \\ \dfrac{\partial L(\mathbf{w},b,\xi_i;\alpha_i,\lambda_i)}{\partial \xi_i} = 0 \;\rightarrow\; 0 \le \alpha_i \le C \end{cases} \tag{4}$$

Then the problem (3) can be reformulated as

$$\begin{aligned} &\max_{\boldsymbol{\alpha}} \sum_i \boldsymbol{\alpha}_i - \frac{1}{2}\sum_{i,j} y_i y_j \boldsymbol{\alpha}_i \boldsymbol{\alpha}_j \varphi(\mathbf{x}_i)^{\mathrm{T}} \varphi(\mathbf{x}_j) \\ &s.t. \sum_{i=1}^{N} \alpha_i y_i = 0,\ 0 \le \alpha_i \le C \end{aligned} \tag{5}$$

By solving α of the dual problem (5) with a quadratic programming, the goal of SVM is to construct the following decision function (classifier),

$$f(\mathbf{x}) = \mathrm{sgn}\left(\sum_{i=1}^{M} \boldsymbol{\alpha}_i y_i \kappa(\mathbf{x}_i,\mathbf{x}) + b\right) \tag{6}$$

where $\kappa(\cdot)$ is a kernel function. $\kappa(\mathbf{x}_i,\mathbf{x}) = \varphi(\mathbf{x}_i)^{\mathrm{T}}\varphi(\mathbf{x}) = \mathbf{x}_i^{\mathrm{T}}\mathbf{x}$ for linear SVM and $\kappa(\mathbf{x}_i,\mathbf{x}) = \exp(-\|\mathbf{x}_i - \mathbf{x}\|^2/\sigma^2)$ for RBF-SVM.

2.2. Least Square Support Vector Machine (LSSVM)

LSSVM is an improved and simplified version of SVM. The details can be referred to [20]. We briefly introduce the basic principle of LSSVM. By introducing a square error and an equality constraint in Eq.(2), LSSVM can be formulated as

$$\begin{aligned} &\min_{\mathbf{w},\xi_i} \frac{1}{2}\|\mathbf{w}\|^2 + C\cdot\frac{1}{2}\sum_{i=1}^{N}\xi_i^2, \\ &s.t.\ y_i\left[\mathbf{w}^{\mathrm{T}}\varphi(\mathbf{x}_i) + b\right] = 1 - \xi_i,\ i = 1,\ldots,N \end{aligned} \tag{7}$$

The Lagrange function of (7) can be defined as

$$L(\mathbf{w},b,\xi_i;\alpha_i)=\frac{1}{2}\|\mathbf{w}\|^2+C\cdot\frac{1}{2}\sum_{i=1}^{N}\xi_i^2 - \sum_{i=1}^{N}\alpha_i\left(y_i\left[\mathbf{w}^{\mathrm{T}}\varphi(\mathbf{x}_i)+b\right]-1+\xi_i\right) \tag{8}$$

where α_i is the Lagrange multiplier.

The optimality conditions can be obtained by computing the partial derivatives of (8) with respect to the four variables as

$$\begin{cases} \dfrac{\partial L(\mathbf{w},b,\xi_i;\alpha_i)}{\partial \mathbf{w}}=0 \;\rightarrow\; \mathbf{w}=\sum_{i=1}^{N}\alpha_i y_i\varphi(\mathbf{x}_i) \\ \dfrac{\partial L(\mathbf{w},b,\xi_i;\alpha_i)}{\partial b}=0 \;\rightarrow\; \sum_{i=1}^{N}\alpha_i y_i=0 \\ \dfrac{\partial L(\mathbf{w},b,\xi_i;\alpha_i)}{\partial \xi_i}=0 \;\rightarrow\; \alpha_i=C\xi_i \\ \dfrac{\partial L(\mathbf{w},b,\xi_i;\alpha_i)}{\partial \alpha_i}=0 \;\rightarrow\; y_i\left[\mathbf{w}^{\mathrm{T}}\varphi(\mathbf{x}_i)+b\right]-1+\xi_i=0 \end{cases} \tag{9}$$

The Eq. (9) can be written compactly as

$$\begin{bmatrix} \mathbf{I} & 0 & 0 & -\mathbf{Z}^T \\ 0 & 0 & 0 & -\mathbf{Y}^T \\ 0 & 0 & C\mathbf{I} & -\mathbf{I} \\ \mathbf{Z} & \mathbf{Y} & \mathbf{I} & 0 \end{bmatrix}\begin{bmatrix} \mathbf{w} \\ \mathbf{b} \\ \boldsymbol{\xi} \\ \boldsymbol{\alpha} \end{bmatrix}=\begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \vec{\mathbf{1}} \end{bmatrix} \tag{10}$$

where $\mathbf{Z}=[\varphi(\mathbf{x}_1)y_1,\ldots,\varphi(\mathbf{x}_N)y_N]^{\mathrm{T}}$, $\mathbf{Y}=[y_1,\ldots,y_N]^{\mathrm{T}}$, $\vec{\mathbf{1}}=[1,\ldots,1]^{\mathrm{T}}$, $\boldsymbol{\xi}=[\xi_1,\ldots,\xi_N]^{\mathrm{T}}$, $\boldsymbol{\alpha}=[\alpha_1,\ldots,\alpha_N]^{\mathrm{T}}$.

The solution of $\boldsymbol{\alpha}$ and $\mathbf{b}$ can also be given by

$$\begin{bmatrix} \mathbf{0} & -\mathbf{Y}^{\mathrm{T}} \\ \mathbf{Y} & \mathbf{Z}\mathbf{Z}^{\mathrm{T}}+C^{-1}\mathbf{I} \end{bmatrix}\begin{bmatrix} \mathbf{b} \\ \boldsymbol{\alpha} \end{bmatrix}=\begin{bmatrix} \mathbf{0} \\ \vec{\mathbf{1}} \end{bmatrix} \tag{11}$$

Let $\boldsymbol{\Omega}=\mathbf{Z}\mathbf{Z}^{\mathrm{T}}$, with Mercer condition, there is

$$\Omega_{k,l}=y_k y_l\varphi(\mathbf{x}_k)^{\mathrm{T}}\varphi(\mathbf{x}_l)=y_k y_l\kappa(\mathbf{x}_k,\mathbf{x}_l),\ k,l=1,\ldots,N \tag{12}$$

By substituting (12) into (11), the solution can be obtained by solving a linear equation instead of a quadratic programming problem as SVM. The final decision function of LSSVM is the same as Eq.(6).

2.3. Extreme Learning Machine (ELM)

ELM aims to solve the output weights of a single layer feed-forward neural network (SLFN) by minimizing the squared loss of predicted errors and the norm of the output weights in both classification and regression problems. We briefly introduce the principle of ELM for classification problems. Given a dataset $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N] \in \Re^{d \times N}$ of N samples with label $\mathbf{T} = [\mathbf{t}_1, \mathbf{t}_2, \cdots, \mathbf{t}_N] \in \Re^{c \times N}$, where d is the dimension of sample and c is the number of classes. Note that if $\mathbf{x}_i$ $(i = 1, \cdots, N)$ belongs to the k-th class, the k-th position of $\mathbf{t}_i$ $(i = 1, \cdots, N)$ is set as 1, and -1 otherwise. The hidden layer output matrix $\mathbf{H}$ with L hidden neurons can be computed as

$$\mathbf{H} = \begin{bmatrix} h\left(\mathbf{w}_1^{\mathrm{T}}\mathbf{x}_1 + b_1\right) & h\left(\mathbf{w}_2^{\mathrm{T}}\mathbf{x}_1 + b_2\right) & \cdots & h\left(\mathbf{w}_L^{\mathrm{T}}\mathbf{x}_1 + b_L\right) \\ \vdots & \vdots & & \vdots \\ h\left(\mathbf{w}_1^{\mathrm{T}}\mathbf{x}_N + b_1\right) & h\left(\mathbf{w}_2^{\mathrm{T}}\mathbf{x}_N + b_2\right) & \cdots & h\left(\mathbf{w}_L^{\mathrm{T}}\mathbf{x}_N + b_L\right) \end{bmatrix} \tag{13}$$

where $h(\cdot)$ is the activation function of hidden layer, $\mathbf{W} = [\mathbf{w}_1, \cdots, \mathbf{w}_L] \in \Re^{d \times L}$ and $\mathbf{B} = [b_1, \cdots, b_L]^{\mathrm{T}} \in \Re^{L}$ are randomly generated input weights and bias between the input layer and hidden layer. With such a hidden layer output matrix $\mathbf{H}$, ELM can be formulated as follows

$$\begin{aligned} &\min_{\boldsymbol{\beta} \in \Re^{L \times c}} \frac{1}{2}\|\boldsymbol{\beta}\|^2 + C \cdot \frac{1}{2}\sum_{i=1}^{N}\|\boldsymbol{\xi}_i\|^2 \\ &s.t.\, h(\mathbf{x}_i)\boldsymbol{\beta} = \mathbf{t}_i^{\mathrm{T}} - \boldsymbol{\xi}_i^{\mathrm{T}}, i = 1, \ldots, N \Leftrightarrow \mathbf{H}\boldsymbol{\beta} = \mathbf{T}^{\mathrm{T}} - \boldsymbol{\xi}^{\mathrm{T}} \end{aligned} \tag{14}$$

where $\boldsymbol{\beta} \in \Re^{L \times c}$ denotes the output weights between hidden layer and output layer, $\boldsymbol{\xi} = [\boldsymbol{\xi}_1, \cdots, \boldsymbol{\xi}_N]$ denotes the prediction error matrix with respect to the training data, and C is a penalty constant on the training errors.

The closed form solution $\boldsymbol{\beta}$ of (14) can be easily solved as

$$\boldsymbol{\beta}^* = \mathbf{H}^{+}\mathbf{T} = \begin{cases} \left(\mathbf{H}^{\mathrm{T}}\mathbf{H} + \dfrac{\mathbf{I}_{L \times L}}{C}\right)^{-1}\mathbf{H}^{\mathrm{T}}\mathbf{T}, \text{ if } N \geq L \\ \mathbf{H}^{\mathrm{T}}\left(\mathbf{H}\mathbf{H}^{\mathrm{T}} + \dfrac{\mathbf{I}_{N \times N}}{C}\right)^{-1}\mathbf{T}, \text{if } N < L \end{cases} \tag{15}$$

where $\mathbf{I}_{L \times L}$ denotes the identity matrix with size of L, and $\mathbf{H}^{+}$ is the Moore-Penrose generalized inverse of $\mathbf{H}$.

Then the predicted output of a new observation $\mathbf{z}$ can be computed as

$$\mathbf{y} = h(\mathbf{z})\boldsymbol{\beta}^* = \begin{cases} h(\mathbf{z})\cdot\left(\mathbf{H}^{\mathrm{T}}\mathbf{H} + \frac{\mathbf{I}_{L\times L}}{C}\right)^{-1}\mathbf{H}^{\mathrm{T}}\mathbf{T}, & if\ N \geq L \\ h(\mathbf{z})\cdot\mathbf{H}^{\mathrm{T}}\left(\mathbf{H}\mathbf{H}^{\mathrm{T}} + \frac{\mathbf{I}_{N\times N}}{C}\right)^{-1}\mathbf{T}, & if\ N < L \end{cases} \tag{16}$$

2.4. Kernelized Extreme Learning Machine (KELM)

With Mercer condition, a KELM is formulated and described as follows. Let $\boldsymbol{\Omega} = \mathbf{H}\mathbf{H}^{\mathrm{T}} \in \Re^{N\times N}$, where $\Omega_{i,j} = h(\mathbf{x}_i)h(\mathbf{x}_j)^{\mathrm{T}} = \kappa(\mathbf{x}_i, \mathbf{x}_j)$ and $\kappa(\cdot)$ is the kernel function. With the expression of solution $\boldsymbol{\beta}$ (15), the predicted output of a new observation $\mathbf{z}$ can be computed as

$$\begin{aligned} \mathbf{y} &= h(\mathbf{z})\boldsymbol{\beta}^* \\ &= h(\mathbf{z})\cdot\mathbf{H}^{\mathrm{T}}\left(\mathbf{H}\mathbf{H}^{\mathrm{T}} + \frac{\mathbf{I}_{N\times N}}{C}\right)^{-1}\mathbf{T} \\ &= \begin{bmatrix} \kappa(\mathbf{z}, \mathbf{x}_1) \\ \vdots \\ \kappa(\mathbf{z}, \mathbf{x}_1) \end{bmatrix}^{\mathrm{T}}\left(\boldsymbol{\Omega} + \frac{\mathbf{I}_{N\times N}}{C}\right)^{-1}\mathbf{T} \end{aligned} \tag{17}$$

Note that due to the kernel matrix of training data is $\boldsymbol{\Omega} \in \Re^{N\times N}$, therefore, the number L of hidden neurons is not explicit and the decision function of KELM can be expressed uniquely as (17).

3. Proposed Adaptive Extreme Learning Machine

3.1. Notations

In this paper, the subscript 'S' and 'T' represent the source and target domains, respectively. $\mathbf{X}_S = \left[\mathbf{x}_S^1, \cdots, \mathbf{x}_S^{N_S}\right] \in \Re^{d\times N_S}$ and $\mathbf{X}_T = \left[\mathbf{x}_T^1, \cdots, \mathbf{x}_T^{N_T}\right] \in \Re^{d\times N_T}$ represent the training set of source data and labeled target data. $\mathbf{Y}_S = \left[\mathbf{y}_S^1, \cdots, \mathbf{y}_S^{N_S}\right] \in \Re^{C\times N_S}$ and $\mathbf{Y}_T = \left[\mathbf{y}_T^1, \cdots, \mathbf{y}_T^{N_T}\right] \in \Re^{C\times N_T}$ denote the label matrix of two tasks. N_S and N_T denote the number of source data and labeled target data, respectively. C and d indicate the number of classes and dimensions, respectively. We define $\mathbf{X}_U$ as the unlabeled data matrix of target domain. $\|\cdot\|_{\mathrm{F}}$ denotes Frobenious norm, $\|\cdot\|_2$ denotes l_2- norm, $Tr(\cdot)$ denotes trace operator, and $(\cdot)^{\mathrm{T}}$ denotes transpose of matrix or vector. Throughout the paper, matrix is represented by uppercase boldface, vector is represented by lowercase boldface, and variable is shown in italics.

3.2. Problem Formulation

The traditional ELM aims to solve a single task learning problem. In this paper, cross task learning problem is expected to be solved by using extreme learning machine. Therefore, for adapting the traditional

ELM in cross-task learning, a new adaptive extreme learning machine (AELM) method is introduced. The generalized formulation of the proposed AELM can be shown as the following minimization problem.

$$\min_{\boldsymbol{\beta}} \frac{1}{2}\|\boldsymbol{\beta}\|_{\mathrm{F}}^2 + \lambda \cdot E_S(\boldsymbol{\beta}) + \mu \cdot E_T(\boldsymbol{\beta}) + \Omega(\boldsymbol{\beta}) \tag{18}$$

where the first term denotes the regularization of AELM classifier hyper-parameters, the second term denotes the prediction error of source domain, the third term denotes the prediction error of labeled data in target domain (new task), and the last term denotes the local structure preservation term accounting for the unlabeled data in target domain. Specifically, the construction process of model (18) is shown as follows.

As can be seen from ELM in Eq.(14), the error term $E_S(\boldsymbol{\beta})$ is calculated based on the training data of single task (i.e. source domain). Therefore, $E_S(\boldsymbol{\beta})$ can be formulated as follows

$$\min_{\boldsymbol{\beta},\,\xi_T} E_S(\boldsymbol{\beta}) = \frac{1}{2}\sum_{i=1}^{N_S}\left\|\xi_S^i\right\|_2^2, \;\; s.t.\left(\xi_S^i\right)^{\mathrm{T}} = h\left(\mathbf{x}_S^i\right)\boldsymbol{\beta} - \left(\mathbf{y}_S^i\right)^{\mathrm{T}}, i = 1,\ldots,N_S \tag{19}$$

However, there is no error term accounting for the new task (i.e. target domain). Therefore, we introduce a new error term for target domain, by supposing that there is a very few labeled data available in target domain. The minimization problem of the new error term $E_T(\boldsymbol{\beta})$ is formulated as follows

$$\min_{\boldsymbol{\beta},\,\xi_T} E_T(\boldsymbol{\beta}) = \frac{1}{2}\sum_{j=1}^{N_T}\left\|\xi_T^j\right\|_2^2, \;\; s.t.\left(\xi_T^j\right)^{\mathrm{T}} = h\left(\mathbf{x}_T^j\right)\boldsymbol{\beta} - \left(\mathbf{y}_T^j\right)^{\mathrm{T}}, \;\; j = 1,\ldots,N_T \tag{20}$$

By incorporating the error term Eq.(19) and Eq.(20) together, we can obtain the following model

$$\begin{aligned} &\min_{\boldsymbol{\beta}\in\Re^{L\times C}} \frac{1}{2}\|\boldsymbol{\beta}\|^2 + \lambda\cdot\frac{1}{2}\sum\nolimits_{i=1}^{N_S}\left\|\xi_S^i\right\|_2^2 + \mu\cdot\frac{1}{2}\sum_{j=1}^{N_T}\left\|\xi_T^j\right\|_2^2 \\ &s.t. \;\; \left(\xi_S^i\right)^{\mathrm{T}} = h\left(\mathbf{x}_S^i\right)\boldsymbol{\beta} - \left(\mathbf{y}_S^i\right)^{\mathrm{T}}, \;\; i = 1,\ldots,N_S \\ &\qquad \left(\xi_T^j\right)^{\mathrm{T}} = h\left(\mathbf{x}_T^j\right)\boldsymbol{\beta} - \left(\mathbf{y}_T^j\right)^{\mathrm{T}}, \;\; j = 1,\ldots,N_T \end{aligned} \tag{21}$$

Furthermore, as described in Eq.(20), the error term of the new task is calculated based on only a few labeled data in target domain. However, the unlabeled data in target domain is also useful for cross-task recognition. Therefore, based on the unsupervised graph manifold learning that has local structure preservation capability, a manifold regularization term $\Omega(\boldsymbol{\beta})$ based on Laplacian graph is used in the proposed AELM model, such that the intrinsic geometric structural information of a number of unlabeled data in the new task (i.e. target domain) can be well exploited. The manifold assumption is that if two data

points are close to each other in the intrinsic data geometry, their associated label prediction of ELM should also be close [34, 35]. With the manifold regularization, semi-supervised cross-task learning of ELM can be achieved. Generally, the Laplacian graph based manifold regularization term $\Omega(\boldsymbol{\beta})$ is formulated as

$$\min_{\boldsymbol{\beta}} \Omega(\boldsymbol{\beta}) = \mathbf{F}^{\mathrm{T}}\mathbf{L}\mathbf{F} \tag{22}$$

where $\mathbf{F}$ denotes the predicted label matrix (output) of the unlabeled data and can be represented as

$$\mathbf{F} = h(\mathbf{X}_U)\boldsymbol{\beta} \tag{23}$$

In Eq.(22), $\mathbf{L}$ is the well-known Laplacian matrix, which can be calculated as $\mathbf{L} = \mathbf{D} - \mathbf{W}$. The $\mathbf{D}$ is a diagonal matrix, and its entry $D_{ii} = \sum_j W_{i,j}$. The entry of matrix $\mathbf{W}$ is generally calculated as follows

$$W_{i,j} = \begin{cases} 1, \text{ if } \mathbf{x}_i \in N_k(\mathbf{x}_j) \text{or } \mathbf{x}_j \in N_k(\mathbf{x}_i) \\ 0, \text{otherwise} \end{cases} \tag{24}$$

where $N_k(\mathbf{x}_i)$ denotes the set of k nearest neighbors of $\mathbf{x}_i$.

By substituting Eq.(23) into Eq.(22), the manifold regularization term $\Omega(\boldsymbol{\beta})$ can be further formulated as

$$\min_{\boldsymbol{\beta}} \Omega(\boldsymbol{\beta}) = \frac{1}{2}\boldsymbol{\beta}^{\mathrm{T}} h(\mathbf{X}_U)^{\mathrm{T}} \mathbf{L} h(\mathbf{X}_U)\boldsymbol{\beta} \tag{25}$$

By substituting the three terms of Eq.(19), Eq.(20) and Eq.(25) into Eq.(18), the proposed AELM can be illustrated as

$$\begin{aligned} &\min_{\boldsymbol{\beta}\in\Re^{L\times C}} \frac{1}{2}\|\boldsymbol{\beta}\|^2 + \lambda\cdot\frac{1}{2}\sum_{i=1}^{N_S}\left\|\boldsymbol{\xi}_S^i\right\|_2^2 + \mu\cdot\frac{1}{2}\sum_{j=1}^{N_T}\left\|\boldsymbol{\xi}_T^j\right\|_2^2 + \frac{1}{2}\boldsymbol{\beta}^{\mathrm{T}} h(\mathbf{X}_U)^{\mathrm{T}}\mathbf{L}h(\mathbf{X}_U)\boldsymbol{\beta} \\ &s.t.\ \left(\boldsymbol{\xi}_S^i\right)^{\mathrm{T}} = h\left(\mathbf{x}_S^i\right)\boldsymbol{\beta} - \left(\mathbf{y}_S^i\right)^{\mathrm{T}},\ \ i = 1,\ldots,N_S \\ &\left(\boldsymbol{\xi}_T^j\right)^{\mathrm{T}} = h\left(\mathbf{x}_T^j\right)\boldsymbol{\beta} - \left(\mathbf{y}_T^j\right)^{\mathrm{T}},\ \ j = 1,\ldots,N_T \end{aligned} \tag{26}$$

where λ and μ represent the trade-off coefficients.

Finally, the proposed AELM model (26) can be compactly written as

$$\begin{aligned} &\min_{\boldsymbol{\beta}\in\Re^{L\times C}} \frac{1}{2}\|\boldsymbol{\beta}\|^2 + \lambda\cdot\frac{1}{2}\|\boldsymbol{\xi}_S\|_{\mathrm{F}}^2 + \mu\cdot\frac{1}{2}\|\boldsymbol{\xi}_T\|_{\mathrm{F}}^2 + \frac{1}{2}\boldsymbol{\beta}^{\mathrm{T}}\mathbf{H}_U^{\mathrm{T}}\mathbf{L}\mathbf{H}_U\boldsymbol{\beta} \\ &s.t.\ \boldsymbol{\xi}_S^{\mathrm{T}} = \mathbf{H}_S\boldsymbol{\beta} - \mathbf{Y}_S^{\mathrm{T}},\ \ i = 1,\ldots,N_S \\ &\boldsymbol{\xi}_T^{\mathrm{T}} = \mathbf{H}_T\boldsymbol{\beta} - \mathbf{Y}_T^{\mathrm{T}},\ \ j = 1,\ldots,N_T \end{aligned} \tag{27}$$

where $\mathbf{H}_S$, $\mathbf{H}_T$, and $\mathbf{H}_U$ denote the hidden layer matrix of the source training data, labeled target training data and unlabeled target training data, respectively. The hidden layer activation function is $h(\cdot)$. $\boldsymbol{\xi}_S$ and $\boldsymbol{\xi}_T$ represent the prediction error matrix of source and labeled target training data.

3.3. Problem Solver

The proposed AELM in Eq.(27) can be easily solved by using similar solver with ELM. The AELM solver can be discussed as two cases. In this paper, the solver can be determined according to the size of N_S and L (i.e., the number of hidden nodes).

When $N_S < L$, $\mathbf{H}_S$ has more columns than rows and of full row rank, the problem becomes a under-determined least square problem. Therefore the solver of classifier $\boldsymbol{\beta}$ in AELM can be easily solved as

$$\boldsymbol{\beta} = \left(\mathbf{I} + \mathbf{H}_U^{\mathrm{T}}\mathbf{L}\mathbf{H}_U\right)^{-1}\left(\begin{array}{l}\mathbf{H}_S^{\mathrm{T}}\left(\mathbf{V}\mathbf{R}^{-1}\mathbf{Q} - \mathbf{W}\right)^{-1}\left(\mathbf{V}\mathbf{R}^{-1}\mathbf{Y}_T - \mathbf{Y}_S\right) + \cdots \\ \qquad \mathbf{H}_T^{\mathrm{T}}\left(\mathbf{R}^{-1}\mathbf{Y}_T - \mathbf{R}^{-1}\mathbf{Q}\left(\mathbf{V}\mathbf{R}^{-1}\mathbf{Q} - \mathbf{W}\right)^{-1}\left(\mathbf{V}\mathbf{R}^{-1}\mathbf{Y}_T - \mathbf{Y}_S\right)\right)\end{array}\right) \tag{28}$$

where $\mathbf{Q} = \mathbf{H}_T\mathbf{H}_S^{\mathrm{T}}$, $\mathbf{R} = \mathbf{H}_T\mathbf{H}_T^{\mathrm{T}} + \mathbf{I}/\mu$, $\mathbf{V} = \mathbf{H}_S\mathbf{H}_T^{\mathrm{T}}$, and $\mathbf{W} = \mathbf{H}_S\mathbf{H}_S^{\mathrm{T}} + \mathbf{I}/\lambda$.

When $N_S > L$, $\mathbf{H}_S$ has more rows than columns and is of full column rank, the problem becomes an over-determined least square problem. The solver of classifier $\boldsymbol{\beta}$ in AELM can be easily solved as

$$\boldsymbol{\beta} = \left(\mathbf{I} + \lambda \cdot \mathbf{H}_S^{\mathrm{T}}\mathbf{H}_S + \mu \cdot \mathbf{H}_T^{\mathrm{T}}\mathbf{H}_T + \mathbf{H}_U^{\mathrm{T}}\mathbf{L}\mathbf{H}_U\right)^{-1}\left(\lambda \cdot \mathbf{H}_S^{\mathrm{T}}\mathbf{Y}_S + \mu \cdot \mathbf{H}_T^{\mathrm{T}}\mathbf{Y}_T\right) \tag{29}$$

The complete algorithm of the proposed AELM is summarized in **Algorithm 1**.

Algorithm 1. AELM

Input: $\mathbf{X}_S$, $\mathbf{X}_T$, $\mathbf{X}_U$, $\mathbf{Y}_S$, $\mathbf{Y}_T$, L, λ, μ

Procedure:

Step 1. ELM network initialization with random input weights $\mathbf{W}$ and bias $\mathbf{B}$;

Step 2. Calculate the hidden layer output matrix $\mathbf{H}_S = h(\mathbf{W}\mathbf{X}_S + \mathbf{B})$, $\mathbf{H}_T = h(\mathbf{W}\mathbf{X}_T + \mathbf{B})$, $\mathbf{H}_U = h(\mathbf{W}\mathbf{X}_U + \mathbf{B})$;

Step 3. Calculate the Laplacian matrix $\mathbf{L}$ according to Eq.(24);

Step 4. Calculate the output weights (classifier) $\boldsymbol{\beta}$ by using Eq.(28) or Eq.(29).

Output: $\boldsymbol{\beta}$

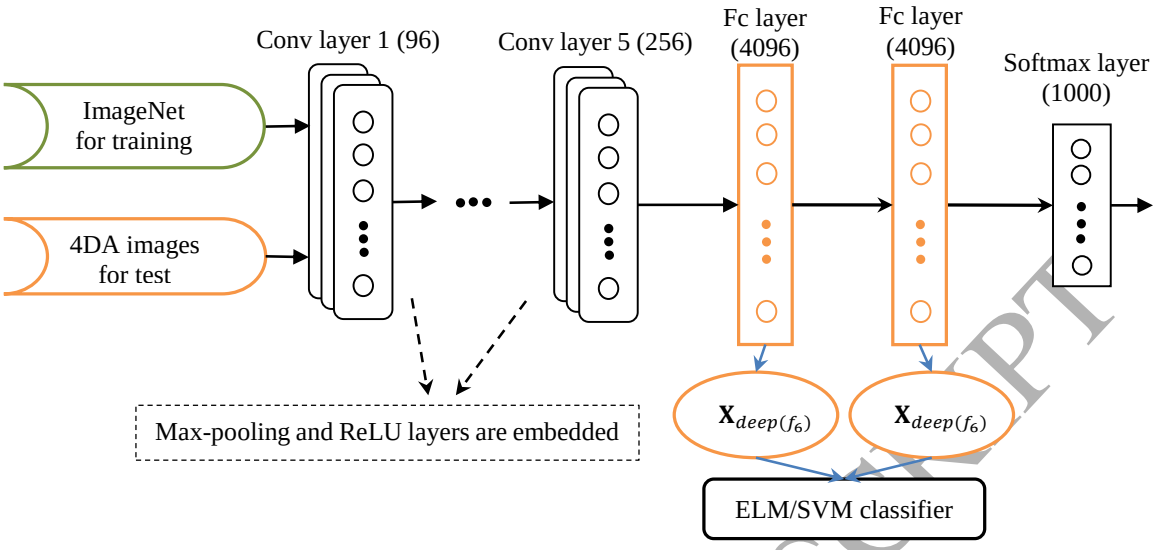

Fig. 2. Diagram of AlexNet based training and testing protocol in this paper

4. ELMs/SVMs plus Convolutional Neural Network

4.1. Deep Learning on ImageNet based on CNN

In this paper, we aim at proposing a comparative investigation on SVMs and ELMs for classification based on deep convolutional features. Therefore, we adopt the deep convolutional activated features (DeCAF) from [17] for experiments. The structures of CNN for training on the ImageNet with 1000 categories are the same as the proposed CNN in [10]. The basic structure of the adopted net (AlexNet) is illustrated in Fig.2, which includes 5 convolutional layers and 3 fully-connected layers. The first convolutional layer filters the 224×224×3 input image with 96 kernels with size 11×11×3. The input of the second convolutional (conv) layer is feed with the normalized and pooled output of the first conv layer, and it filters the input with 256 kernels with size 5×5×48. The third conv layer has 384 kernels with size 3×3×256 which is connected to the normalized and pooled output of the second conv layer. The fourth conv layer has 384 kernels with size 3×3×192, and the fifth conv layer has 256 kernels with size 3×3×192. The fully-connected layers have 4096 neurons, respectively. Note that each conv layer follows a 'max-pooling' layer. Stochastic gradient descent is a better choice for training. Many further details of the CNN training architecture and network parameters can be referred to [10].

Fig. 3. Examples of object images from three sources: Amazon (1st row), DSLR (2nd row), Webcam (3rd row) and Caltech 256 (4th row). Different visual cues such as camera viewpoint, resolution, illumination, and background have been well illustrated.

Table 1 Details of 4DA-CNN Datasets

Dataset	#class	#dimension	#samples	n_s/c	n_t/c
Amazon	10	4096	958	20	3
DSLR	10	4096	157	8	3
Webcam	10	4096	295	8	3
Caltech	10	4096	1123	8	3

4.2. Deep Representation of 4DA Benchmark Datasets based on Pre-trained CNN

The well-trained network parameters shown in Fig.2 are used for deep representation of the 4DA (domain adaptation) dataset [31, 32]. The CNN outputs of the 6-th (f_6) and 7-th (f_7) fully-connected layers are used as inputs of SVMs and ELMs for classification, respectively. The 4DA dataset includes four domains such as Caltech 256 (C), Amazon (A), Webcam (W) and Dslr (D) sampled from different sources, in which 10 object classes are selected. As can be seen from Fig.2, the dimension of features from f_6 and f_7 is 4096. The detail of 4DA dataset with deep features is summarized in Table 1. Some examples of the dataset for each domain have been illustrated in Fig.3.

4.3. Classification

The 4DA dataset is commonly used for evaluating domain adaptation and transfer learning tasks. So, in this paper, we investigate the classification ability of deep representation on domain shifted data. We adopt the deep features for SVMs/ELMs training, and compare the classification accuracy. The specific experimental setup is described in Experiments section.

5. Experiments

5.1. Experimental Setup

In the experiment, three settings are investigated respectively, as follows.

1) **Setting 1:** *single-domain* recognition task.

For example, we train a model on the training data of Amazon, and report the test accuracy on the remaining data of Amazon. As shown in Table 1 (n_s/c), 20, 8, 8, and 8 samples per class are randomly selected for training from Amazon, DSLR, Webcam and Caltech domains, respectively, and the remaining are used as test samples for each domain. 20 random train/test splits are run, and the average recognition accuracy for each method is reported.

2) **Setting 2:** *cross-domain* recognition tasks (source only).

We perform a cross-domain recognition task. For example, we train a SVM/ELM on the Amazon and test on DSLR, i.e. A→D. Totally, 12 cross-domain tasks among the four domains are conducted. Note that the training data is source data only (source only) without leveraging the data from target domain. The number of training data is 20, 8, 8 and 8 per class for Amazon, DSLR, Webcam and Caltech domains, respectively, when used as source domain. 20 random train/test splits are run, and the average recognition accuracy for each method is reported.

3) **Setting 3:** *cross-domain* recognition tasks (both source and target).

Similar to Setting 2, we perform a cross-domain recognition task. For example, we train a SVM/ELM on the Amazon and test on DSLR, i.e. A→D. Totally, 12 cross-domain tasks among the four domains are conducted. However, the difference from Setting 2 lies in that the training data includes the labeled source data and few labeled target data. The number of training data is 20, 8, 8 and 8 per class for Amazon, DSLR, Webcam and Caltech domains, respectively, when used as source domain. The number of few labeled target

data is 3 per class for each domain when they are used as target domain, as shown in Table 1 (n_t/c). 20 random train/test splits are run, and the average recognition accuracy for each method is reported.

5.2. Parameter Setting

To make sure that the best result of each method can be obtained, we have adjusted the parameters. According to the experiments, for SVM the penalty coefficient C and kernel parameter σ are set as 1000 and 1, respectively, by using *Libsvm-3.12* toolbox. For LSSVM, the two coefficients are automatically optimized with a grid search by using *LSSVM-1.7* toolbox. For ELM, the penalty coefficient C and the number L of hidden neurons are set as 100 and 5000, respectively. For KELM, the penalty coefficient C and kernel parameter σ are set as 100 and 0.01, respectively. Note that the penalty coefficient C and kernel parameter σ for SVM, ELM, and KELM are adjusted from the set C={1, 100, 10000} and σ={0.0001, 0.01, 1, 100}. The parameter λ and μ of AELM follow the same tuning as C and σ. In ELMs, the *radbas* kernel function is used.

5.3. Experimental Results

*5.3.1. Results based on **Setting 1***

For experimental **Setting 1**, the average accuracy and standard deviation of 20 randomly generated train/test splits for six methods including NN, SVM, LSSVM, ELM, KELM and the proposed AELM are reported in Table 2. We can observe that the recognition performance based on the deep features from the 6-th layer (f_6) and 7-th layer (f_7) is slightly different. The best two methods are highlighted with bold face. From the comparisons, we can find that ELMs outperforms SVMs and NN methods for all domains, and KELM and AELM shows a more competitive performance. Specifically, by comparing KELM and SVM, the improvement in accuracy for the deep features f_6 is 0.8%, 0.2%, 1.1% and 2.1% for Amazon, DSLR, Webcam, and Caltech, respectively. For the deep features f_7, the improvement is 1.0%, 0.6%, 0.8%, and 2.5%, respectively.

Table 2 Recognition accuracy of each method for different domains in **Setting 1**

Method	CNN_layer	Amazon	DSLR	Webcam	Caltech	CNN_layer	Amazon	DSLR	Webcam	Caltech
NN	f_6	91.0±0.3	97.3±0.6	95.0±0.4	75.0±0.4	f_7	92.4±0.2	96.8±0.5	95.3±0.5	76.2±0.5
SVM	f_6	92.9±0.1	97.6±0.6	96.7±0.3	83.9±0.4	f_7	93.2±0.1	96.9±0.5	96.5±0.4	83.2±0.5
LSSVM	f_6	92.9±0.2	97.5±0.4	96.4±0.4	84.6±0.3	f_7	93.5±0.1	96.3±0.6	95.4±0.4	83.9±0.4
ELM	f_6	92.9±0.1	98.0±0.3	97.7±0.2	84.8±0.3	f_7	93.6±0.1	97.2±0.4	97.4±0.3	85.0±0.3
KELM	f_6	93.7±0.1	97.8±0.3	97.8±0.2	86.0±0.3	f_7	94.2±0.1	97.5±0.4	97.3±0.4	85.7±0.3
AELM	f_6	93.5±0.2	97.9±0.4	97.8±0.3	85.8±0.3	f_7	94.1±0.1	97.8±0.3	97.5±0.3	85.7±0.2

*5.3.2. Results of **Setting 2***

Table 3 presents the average recognition accuracy of 20 randomly generated train/test splits based on the experimental setting 2. Totally, 12 cross-domain recognition tasks are conducted. The first two highest accuracies are highlighted in bold face. We can observe that 1) the recognition performance with deep feature f_7 clearly outperforms that of f_6, which demonstrates the effectiveness of "deep"; 2) the performance of ELM and KELM is significantly better than SVM and LSSVM, the average improvement of 12 tasks of KELM is 4% better than that of SVM. The results demonstrate that for more difficult problems (i.e. cross-domain tasks), the ELM based methods show a more competitive and robust advantage for classification. More obvious, the accuracies by using the five methods for each cross-domain task are illustrated in Fig. 4, from which the superiority of ELMs especially KELM is clearly demonstrated compared with others methods for each tasks under deep features from f_7 and f_6.

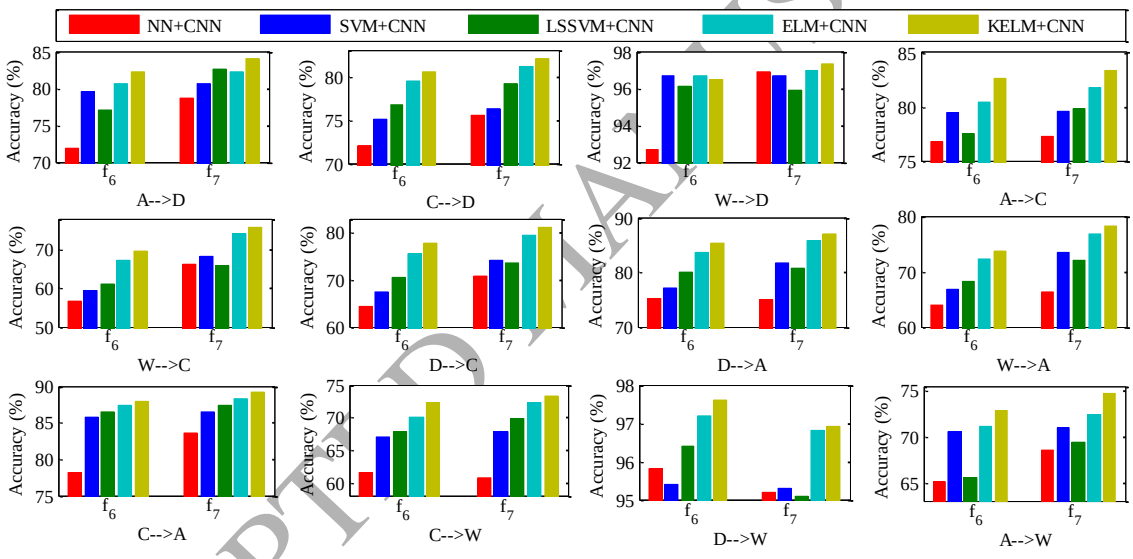

Fig. 4. Recognition accuracies of 12 cross-domain tasks by using NN, SVM, LSSVM, ELM and KELM on the deep convolutional activation features of f_6 and f_7 with experimental **Setting 2**

Table 3 Recognition accuracy of each method with **Setting 2**, where the training data is from source domain only (A: Amazon, C: Caltech 256, W: Webcam, D: Dslr)

Method	CNN_layer	A→D	C→D	W→D	A→C	W→C	D→C	D→A	W→A	C→A	C→W	D→W	A→W
NN	f_6	71.9±0.9	72.0±1.7	92.7±0.5	76.8±0.3	56.6±0.9	64.4±0.4	75.1±0.7	64.0±0.6	78.1±0.8	61.5±1.1	95.8±0.4	65.1±1.0
	f_7	78.7±0.5	75.6±1.3	96.9±0.4	77.2±0.4	66.2±0.5	70.7±0.4	75.0±0.7	66.3±0.8	83.6±0.4	60.7±1.2	95.2±0.4	68.5±0.8
SVM	f_6	79.6±0.7	75.1±1.8	96.7±0.4	79.5±0.4	59.5±0.9	67.3±1.2	77.0±1.0	66.8±1.0	85.8±0.4	67.1±1.1	95.4±0.4	70.6±0.8
	f_7	80.6±0.8	76.4±1.4	96.7±0.4	79.6±0.4	68.1±0.6	74.3±0.6	81.8±0.5	73.4±0.7	86.5±0.5	67.8±1.1	95.3±0.5	71.0±0.8
LSSVM	f_6	77.1±0.9	76.8±1.2	96.1±0.3	77.5±0.6	61.1±0.7	70.6±1.0	80.0±0.8	68.2±1.1	86.5±0.4	67.8±1.2	96.4±0.4	65.5±0.8
	f_7	82.6±0.5	79.2±0.8	95.9±0.4	79.8±0.5	66.0±1.3	73.7±0.9	80.8±0.7	72.0±1.1	87.4±0.3	69.9±1.1	95.1±0.3	69.4±0.6
ELM	f_6	80.6±0.6	79.5±1.2	96.7±0.2	80.4±0.3	67.2±0.5	75.6±0.5	83.7±0.4	72.2±0.9	87.3±0.4	70.1±0.9	97.2±0.3	71.1±0.6
	f_7	82.3±0.5	81.2±0.7	97.0±0.4	81.8±0.3	74.0±0.3	79.5±0.2	85.8±0.3	76.7±0.9	88.3±0.2	72.3±0.9	96.8±0.3	72.4±0.8
KELM	f_6	82.3±0.5	80.7±0.9	96.5±0.3	82.6±0.3	69.5±0.4	77.8±0.4	85.3±0.4	73.8±1.1	88.0±0.4	72.3±1.0	97.6±0.2	72.9±0.7
	f_7	84.0±0.4	82.2±0.9	97.3±0.3	83.4±0.2	75.7±0.3	81.1±0.2	87.1±0.2	78.2±0.8	89.1±0.3	73.3±0.9	96.9±0.3	74.7±0.8
AELM	f_6	83.1±0.3	82.1±0.8	96.6±0.3	82.8±0.2	69.0±0.5	77.7±0.3	84.8±0.4	73.8±1.1	88.6±0.4	72.6±0.9	97.3±0.3	73.5±0.6
	f_7	84.3±0.5	83.7±0.8	97.3±0.3	84.1±0.2	76.6±0.4	81.1±0.3	86.7±0.2	79.4±0.6	89.5±0.4	75.5±0.9	96.7±0.3	76.0±0.7

*5.3.3. Results of **Setting 3**.*

The results under experimental Setting 3 are reported in Table 4, from which we can find that ELMs especially KELM and AELM outperform other methods. Due to that few labeled data from target domain are leveraged in model training with domain adaptation, so the recognition accuracies are much higher than that from Table 3. The average differences between ELMs and SVMs are therefore reduced from 4% in **Setting 2** to 1.5% in **Setting 3**. For better visualization of the difference, we provide a Fig.5 which describes the recognition accuracies of all methods for each cross-domain task. We can see that KELM and AELM always show the best performance.

Table 4 Recognition accuracy of each method with **Setting 3**, where the training data is from both source and target domains (A: Amazon, C: Caltech 256, W: Webcam, D: Dslr)

Method	CNN_layer	A→D	C→D	W→D	A→C	W→C	D→C	D→A	W→A	C→A	C→W	D→W	A→W
NN	f_6	89.4±0.7	90.1±0.8	97.0±0.4	78.1±0.4	69.0±0.9	72.8±0.8	83.8±0.5	83.3±0.7	85.4±0.4	86.9±0.6	97.2±0.4	86.1±0.8
	f_7	93.0±0.5	90.9±0.9	98.6±0.2	78.9±0.4	73.6±0.6	75.6±0.4	86.7±0.5	84.0±0.5	87.9±0.2	87.8±0.9	96.3±0.2	89.1±0.6
SVM	f_6	94.5±0.4	92.9±0.8	99.1±0.2	84.0±0.3	81.7±0.5	83.0±0.3	90.5±0.2	90.1±0.2	90.0±0.2	91.5±0.6	97.9±0.3	90.4±0.8
	f_7	94.0±0.6	92.7±0.8	98.9±0.2	83.4±0.4	81.2±0.4	82.7±0.4	90.9±0.3	90.6±0.2	90.3±0.2	90.6±0.8	98.0±0.2	91.1±0.8
LSSVM	f_6	92.6±0.5	93.1±0.6	98.8±0.2	82.3±0.5	80.7±0.5	82.3±0.4	90.9±0.2	89.7±0.2	90.3±0.1	90.9±0.6	97.8±0.3	87.7±0.8
	f_7	91.9±0.5	92.4±0.8	98.4±0.2	82.9±0.4	81.7±0.3	82.6±0.5	90.9±0.4	90.0±0.2	90.7±0.2	90.4±0.5	97.2±0.3	89.5±0.7
ELM	f_6	94.6±0.5	93.7±0.6	99.2±0.2	83.4±0.3	81.2±0.3	83.5±0.3	91.1±0.2	90.3±0.2	90.5±0.1	91.6±0.7	98.3±0.2	90.5±0.6
	f_7	94.9±0.4	93.0±0.6	99.0±0.2	84.1±0.2	82.2±0.4	84.1±0.2	91.7±0.2	90.8±0.2	90.9±0.1	91.5±0.7	97.9±0.2	91.7±0.7
KELM	f_6	95.7±0.4	94.1±0.6	99.2±0.2	85.0±0.3	83.0±0.3	84.9±0.2	91.9±0.2	90.8±0.2	91.1±0.1	92.2±0.7	98.6±0.2	91.3±0.6
	f_7	95.5±0.4	93.9±0.6	99.1±0.1	85.4±0.3	83.4±0.3	85.3±0.3	92.1±0.2	91.5±0.2	91.5±0.1	91.9±0.6	98.2±0.3	92.2±0.6
AELM	f_6	96.9±0.4	95.8±0.4	99.0±0.1	84.7±0.2	82.2±0.4	83.9±0.2	91.4±0.3	90.3±0.3	90.7±0.1	94.6±0.7	99.2±0.2	95.2±0.6
	f_7	96.1±0.4	94.8±0.7	98.6±0.2	85.1±0.3	83.0±0.4	84.6±0.3	91.4±0.2	90.8±0.3	91.5±0.2	94.2±0.7	98.7±0.2	95.2±0.6

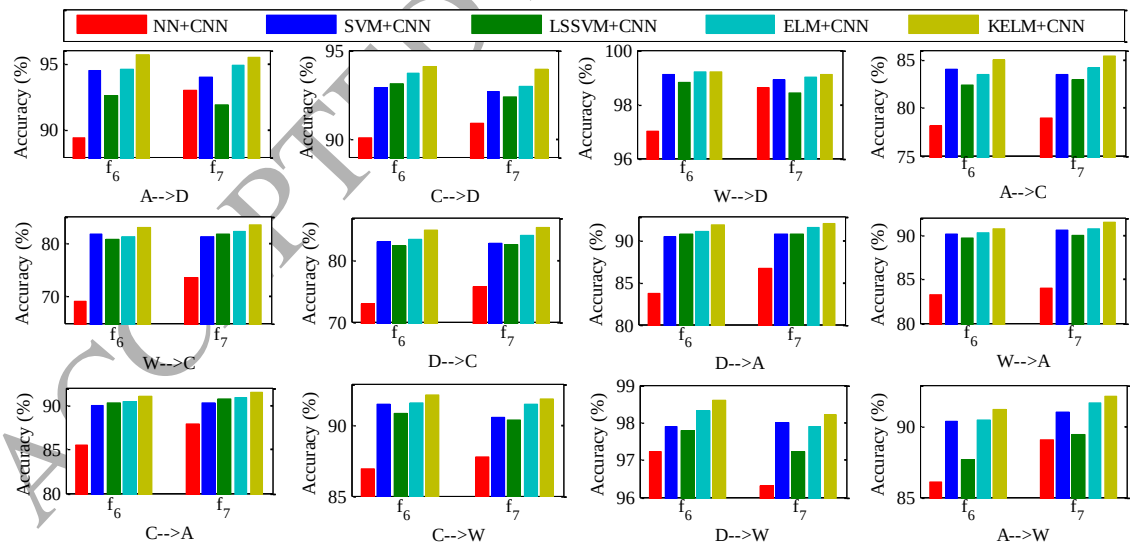

Fig. 5. Recognition accuracies of 12 cross-domain tasks by using NN, SVM, LSSVM, ELM and KELM on the deep convolutional activation features of f_6 and f_7 with experimental **Setting 3**

6. Statistical Significance Analysis

In this paper, for pairwise method comparison, the t-test method is used for statistical significance analysis. In implementation, we have conducted the statistical significance analysis with confidence level of α=0.01 (99%) and α=0.05 (95%) in Table 7, Table 8, and Table 9 on the reported results in Table 2, Table 3, and Table 4, respectively. From the *p*-value and *H*, we can see that there is statistical significance analysis between KELM and other methods with α=0.05. Note that smaller *p*-value or *H*=1 (the null hypothesis is rejected) denotes the significance.

Table 7 Statistical analysis by using t-test on the results of Table 2

Pairwise methods	NN-AELM	SVM-AELM	LSSVM-AELM	ELM-AELM	KELM-AELM
p-value	0.03	0.003	9e-4	0.01	0.85
H (α=0.01)	**0**	1	1	**0**	**0**
H (α=0.05)	1	1	1	1	0

Table 8 Statistical analysis by using t-test on the results of Table 3

Pairwise methods	NN-AELM	SVM-AELM	LSSVM-AELM	ELM-AELM	KELM-AELM
p-value	5e-5	2e-5	9e-5	1e-4	0.01
H (α=0.01)	1	1	1	1	0
H (α=0.05)	1	1	1	1	1

Table 9 Statistical analysis by using t-test on the results of Table 4

Pairwise methods	NN-KELM	SVM-KELM	LSSVM-KELM	ELM-KELM	KELM-AELM
p-value	4e-5	0.001	0.001	0.01	0.4
H (α=0.01)	1	1	1	0	0
H (α=0.05)	1	1	1	1	0

7. Conclusion

In the paper, we present an analysis of ELMs based on object recognition experiments with multiple domains. Motivated by the competitive nature of ELM, an adaptive extreme learning machine (AELM) is proposed for cross-task recognition, which is an extension of traditional ELM from single task to cross-task learning. The very high-level feature learning is trained by CNN on a subset of 1000-category images from ImageNet, and the deep convolutional activation features of the multi-domain objects are used. We aim at exploring the ELM classifiers for high-level deep features in classification, and proposing new ELM approach applicable in multi-task learning and recognition scenarios. In experiments, the deep features of 10-category object images of 4 domains from the 6-th layer and 7-th layer of CNN are used as the inputs of general classifiers, such as NN, SVM, LSSVM, ELM, KELM and AELM. The recognition accuracies for each method under three different experimental settings based on high-level features are reported. A number of experimental results clearly demonstrate that KELM and AELM based classifiers are superior in different

experimental settings. In particular, the proposed AELM shows comparable recognition performance among the presented 5 popular classifiers. The statistical significance test on the recognition accuracies is also implemented and demonstrate the significance with confidence level α=0.05. The experimental analysis demonstrates that ELM plus deep feature representation promotes a competitively effective performance in cross-domain image classification.

Acknowledgement

This work was supported by the National Natural Science Foundation of China under Grant 61401048 and the research fund of central Universities.

References

[1] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86 (1998) 2278-2324.

[2] S. Lawrence, C.L. Giles, T. Ah Chung, and A.D. Back, Face recognition: a convolutional neural-network approach. IEEE Transactions on Neural Networks, 8 (1997) 98-113.

[3] G.E. Hinton and R.R. Salakhutdinov, Reducing the dimensionality of data with neural networks. Science, 313 (2006) 504-507.

[4] G. Hinton, S. Osindero, and Y. The, A fast learning algorithm for deep belief nets. Neural Comput. 18 (2006) 1527-1554.

[5] R. Salakhutdinov and G. Hinton, Deep Boltzman machines, in Proc. Int'l Conf. Artif. Intell. Statist., pp. 448-455, 2009.

[6] G.B. Huang, H. Lee, and E. Learned-Miller, Learning hierarchical representations for face verification with convolutional deep belief networks. Proc. IEEE Int'l Computer Vision and Pattern Recognition, pp. 2518-2525, 2012.

[7] Y. Sun, X. Wang, and X. Tang, Hybrid Deep Learning for Face Verification. Proc. IEEE Int'l Conf. Computer Vision, 2013.

[8] Y. Taigman, M. Yang, M.A. Ranzato, and L. Wolf, DeepFace: Closing the Gap to Human-Level Performance in Face Verification. Proc. IEEE Int'l Computer Vision and Pattern Recognition, 2014.

[9] E. Zhou, Z. Cao, and Q. Yin, Naïve-Deep Face Recognition: Touching the Limit of LFW Benchmark or Not? arXiv: 1501.04690, 2015.

[10] A. Krizhevsky, I. Sutskever, G.E. Hinton, ImageNet classification with deep convolutional neural networks. NIPS, 2012.

[11] D. Ciresan, U. Meier, and J. Schmidhuber, Multi-column deep neural networks for image classification. Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition, pp. 3642-3649, 2012.

[12] A. Karpathy, G. Toderici, S. Shetty, and T. Leung, Large-Scale Video Classification with Convolutional Neural Networks. Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition, pp. 1725-1732, 2014.

[13] A.S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, CNN Features Off-the-Shelf: An Astounding Baseline for Recognition. Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition, pp. 512-519, 2014.

[14] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Accurate Object Detection and Semantic Segmentation. Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition, pp. 580-587, 2014.

[15] K. He, X. Zhang, S. Ren, and J. Sun, Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. arXiv:

1406.4729.

[16] K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun, What is the best multi-stage architecture for object recognition? ICCV, pp. 2146-2153, 2009.

[17] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, T. Darrell, DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition. arXiv: 1310.1531, 2013.

[18] T. Cover and P. Hart, Nearest neighbor pattern classification. IEEE Trans. Information Theory, 13 (1967) 21-27.

[19] V. Vapnik, Statistical learning theory. John Wiley: New York, 1998.

[20] J.A.K. Suykens and J. Vandewalle, Least Squares Support Vector Machine Classifiers, Neural Processing Letters, 9 (1999) 293-300.

[21] G.B. Huang, Q.Y. Zhu, C.K. Siew, Extreme learning machine: Theory and applications. Neurocomputing, 70 (2006) 489-501.

[22] G.B. Huang, H. Zhou, X. Ding, R. Zhang, Extreme Learning Machine for Regression and Multiclass Classification. IEEE Trans. Systems, Man, Cybernetics: Part B, 42 (2012) 513-529.

[23] G.B. Huang, X.J. Ding, H.M. Zhou, Optimization method based on extreme learning machine for classification. Neurocomputing, 74 (2010) 155-163.

[24] L. Zhang and D. Zhang, Evolutionary Cost-sensitive Extreme Learning Machine, IEEE Trans. Neural Networks and Learning Systems, 2016. Doi:10.1109/TNNLS.2016.2607757

[25] G.B. Huang, What are Extreme Learning Machines? Filling the Gap between Frank Rosenblatt's Dream and John von Neumann's Puzzle. Cognitive Computation, 7 (2015) 263-278.

[26] L.L.C. Kasun, H. Zhou, G.B. Huang, and C.M. Vong, Representational Learning with Extreme Learning Machine for Big Data. IEEE Intelligent Systems, 28 (2013) 31-34.

[27] G.-B. Huang, Z. Bai, L. L. C. Kasun, and C. M. Vong, Local Receptive Fields Based Extreme Learning Machine. *IEEE Computational Intelligence Magazine*, 10 (2015) 18-29.

[28] L. Zhang and D. Zhang, LSDT: Latent Sparse Domain Transfer Learning for Visual Adaptation, IEEE Trans. Image Processing, 25(3) (2016) 1177-1191.

[29] L. Zhang, and D. Zhang, Domain Adaptation Extreme Learning Machines for Drift Compensation in E-nose Systems," IEEE Transactions on Instrumentation and Measurement, 64 (2015) 1790-1801.

[30] G. Huang, S. Song, J.N.D. Gupta, and C. Wu, Semi-Supervised and Unsupervised Extreme Learning Machines, IEEE Transactions on Cybernetics, 44 (2014) 2405-2417.

[31] K. Saenko, B. Kulis, M. Fritz, and T. Darrell, Adapting visual category models to new domains, ECCV, 2010.

[32] B. Gong, Y. Shi, F. Sha, and K. Grauman, Geodesic flow kernel for unsupervised domain adaptation, CVPR, pp. 2066-2073, 2012.

[33] Y. Guo, Y. Liu, A. Oerlemans, S. Lao, S. Wu, and M.S. Lew, Deep learning for visual understanding: A review. Neurocomputing, 184 (2016) 27-84.

[34] S. Yan, D. Xu, B. Zhang, H.-J. Zhang, Q. Yang, and S. Lin, Graph embedding and extensions: A general framework for dimensionality reduction, IEEE Trans. Pattern Anal. Mach. Intell., 29(1) (2007) 40-51.

[35] L. Zhang and D. Zhang, Visual Understanding via Multi-Feature Shared Learning with Global Consistency, IEEE Trans. Multimedia, 18(2) (2016) 247-259.

[36] A. Basu, S. Shuo, H. Zhou, M.H. Li, G.B. Huang, Silicon spiking neurons for hardware implementation of extreme learning machines. Neurocomputing, 102 (2013) 125-134.

[37] L.M. Pavelski, M.R. Delgado, C.P. Almeida, R.A. Goncalves, S.M. Venske, Extreme Learning Surrogate Models in Multi-objective Optimization based on Decomposition, Neurocomputing, 180 (2016) 55-67.

[38] L. Zhang, D. Zhang, Robust Visual Knowledge Transfer via Extreme Learning Machine based Domain Adaptation, IEEE Trans. Image Processing, 25 (10) (2016) 4959-4973.

[39] B.Y. Qu, B.F. Lang, J.J. Liang, A.K. Qin, O.D. Crisalle, Two-hidden-layer extreme learning machine for regression and classification. Neurocomputing, 175 (2016) 826-834.

[40] M.D. Tissera, M.D. McDonnell, Deep extreme learning machines: supervised autoencoding architecture for classification. Neurocomputing, 174 (2016) 42-49.

[41] X.D. Li, W.J. Mao, W. Jiang, Extreme learning machine based transfer learning for data classification, Neurocomputing, 174 (2016) 203-210.

[42] Y. Zhang, L. Zhang, and P. Li, A novel biologically inspired ELM-based network for image recognition, Neurocomputing, 174 (2016) 286-298.

[43] Y.Q. Wang, Z.G. Xie, K. Xu, Y. Dou, and Y.W. Lei, An efficient and effective convolutional auto-encoder extreme learning machine network for 3d feature learning, Neurocomputing, 174 (2016) 988-998.

[44] Y. Yang and Q.M.J. Wu, Multilayer Extreme Learning Machine With Subnetwork Nodes for Representation Learning, IEEE Trans. Cybernetics, 46 (11) (2016) 2570-2583.

[45] Y. Yang and Q.M.J. Wu, and Y. Wang, Autoencoder With Invertible Functions for Dimension Reduction and Image Reconstruction, IEE Trans. Systems, Man, and Cybernetics: Systems, 2016. DOI: 10.1109/TSMC.2016.2637279.

[46] H. Liu, J. Qin, F. Sun, and D. Guo, Extreme Kernel Sparse Learning for Tactile Object Recognition, IEEE Trans. Cybernetics, 2016. DOI: 10.1109/TCYB.2016.2614809.

[47] X. Lai, H. Meng, J. Cao, and Z. Lin, A Sequential Partial Optimization Algorithm for Minimax Design of Separable-Denominator 2-D IIR Filters, IEEE Trans. Signal Processing, 65 (4) (2017) 876-887.

Lei Zhang received his Ph.D degree in Circuits and Systems from the College of Communication Engineering, Chongqing University, Chongqing, China, in 2013. He was selected as a Hong Kong Scholar in China in 2013, and worked as a Post-Doctoral Fellow with The Hong Kong Polytechnic University, Hong Kong, from 2013 to 2015. He is currently a Professor/Distinguished Research Fellow with Chongqing University. He has authored more than 50 scientific papers in top journals, including the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, the IEEE TRANSACTIONS ON IMAGE PROCESSING, the IEEE TRANSACTIONS ON MULTIMEDIA, the IEEE TRANSACTIONS ON INSTRUMENTATION AND MEASUREMENT, the IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS: SYSTEMS, the IEEE SENSORS JOURNAL, INFORMATION FUSION, SENSORS & ACTUATORS B, and ANALYTICA CHIMICA ACTA. His current research interests include machine learning, pattern recognition, computer vision, machine olfaction and intelligent systems. Dr. Zhang was a recipient of Outstanding Reviewer Award of Sensor Review Journal in 2016, Outstanding Doctoral Dissertation Award of Chongqing, China, in 2015, Hong Kong Scholar Award in 2014, Academy Award for Youth Innovation of Chongqing University in 2013 and the New Academic Researcher Award for Doctoral Candidates from the Ministry of Education, China, in 2012

Zhenwei He receivced his Bachelor degree in Information Engineering in 2014 from Tianjin University, China. From July 2014 to June 2016, he worked in Chongqing Cable Network Inc. Since September 2016, he is currently studying for his Master degree in Chongqing University. His research interests include deep learning and computer vision.

Yan Liu received her Bachelor degree in Information Engineering in 2014 from Chengdu Polytechnic University, China. Since September 2014, she is currently pursuing a Master degree in Chongqing University. Her research interests include electronic nose and intelligent algorithm.