AN ENHANCED HIERARCHICAL EXTREME LEARNING MACHINE WITH RANDOM SPARSE MATRIX BASED AUTOENCODER

Tianlei Wang[1]*, Xiaoping Lai*[1]*, Jiuwen Cao*[1]*, Chi-Man Vong*[2] *and Badong Chen*[3]

[1]Institute of Information and Control, Hangzhou Dianzi University, China
[2]Faculty of Science and Technology, University of Macau, Macau, China
[3]School of Electronic and Information Engineering, Xi'an Jiaotong University, China

ABSTRACT

Recently, by employing the stacked extreme learning machine (ELM) based autoencoders (ELM-AE) and sparse AEs (SAE), multilayer ELM (ML-ELM) and hierarchical ELM (H-ELM) has been developed. Compared to the conventional stacked AEs, the ML-ELM and H-ELM usually achieve better generalization performance with a significantly reduced training time. However, the ℓ_1-norm based SAE may suffer the overfitting problem and it is unable to provide analytical solution leading to long training time for big data. To alleviate these deficiencies, we propose an enhanced H-ELM (EH-ELM) with a novel random sparse matrix based AE (SMA) in this paper. The contributions are in two aspects, 1) utilizing the random sparse matrix, the sparse features can be obtained; 2) the proposed SMA can provide an analytical solution so that the high computational complexity issue in SAE can be addressed. Experimental results on benchmark datasets show that the proposed EH-ELM achieves a higher recognition rate and a faster training speed than H-ELM and ML-ELM.

Index Terms— Extreme learning machine, Autoencoder, Multilayer perceptron, Random sparse matrix.

1. INTRODUCTION

Feature extraction is vital to data processing as discriminative features help in performance enhancement and computational complexity reduction. As a direct and efficient feature extraction method, stacked autoencoders (AEs) have been developed for representation learning and widely used in deep neural networks (DNNs) [13]. The stacked AEs aim to map inputs to outputs with least possible amount of distortions and learn the data representation by setting the desired outputs as the inputs. However, existing stacked AEs in DNNs usually need a long training time due to the slow convergence of the gradient descent based training methods [13–15].

This work was supported by the National Natural Science Foundation of China under Grants 61573123, 61503104, and U1509205, and by the K. C. Wong Education Foundation and DAAD

Recently, the popular extreme learning machine (ELM) [4–12] based AE has been developed in [1, 2] to achieve a fast feature learning. A multilayer ELM (ML-ELM) that stacks several AE layers has been investigated in [1]. ML-ELM has shown better generalization performance with a lower computational complexity than many DNNs [13–15]. But it is pointed out in [2] that the extracted features in ML-ELM tend to be dense in some applications, which may lead to indistinctive representation of important information of data. Moreover, the simply stacked AEs in ML-ELM may not well exploit the advantage of random feature mapping in ELM. To overcome these shortcomings, a hierarchical ELM (H-ELM) with the ℓ_1-norm based sparse AE (SAE) has been developed [2].

However, the ℓ_1-norm based SAE is unable to provide analytical solution, leading to long training time for big data. It may also suffer the overfitting problem. To address these deficiencies, an enhanced H-ELM (EH-ELM) with a novel random sparse matrix based AE (SMA) is developed in this paper. The contributions are summarized as follows, 1) utilizing the random sparse matrix, the sparse features can be obtained; 2) benefiting from using random sparse matrix, the ℓ_2-norm regularized optimization is formulated in the SMA. The resultant solution can be analytically calculated; 3) by virtue of the SMA, the proposed EH-ELM learns faster than ML-ELM and H-ELM, especially for high-dimensional and large data processing.

Organizations of the rest paper are as follow. Section II gives a brief review on ELM, ELM-SAE and H-ELM. Section III presents the proposed method and Section IV shows the experimental results.

2. REVIEW ON ELM, ELM-SAE AND H-ELM

2.1. ELM, ELM-AE and ELM-SAE

ELM was developed for a single hidden layer feedforward neural network (SLFN) and states that the input weight $\mathbf{W}$ and bias $\mathbf{b}$ can be randomly generated. The training thus becomes solving a linear least-squares (LS) problem [4]. Given a training set $\left\{(\mathbf{x}_i, \mathbf{t}_i)\,|\mathbf{x}_i \in \mathbb{R}^d, \mathbf{t}_i \in \mathbb{R}^m, i = 1, 2, \ldots, N\right\}$ and $(\mathbf{W}_{L\times d}, \mathbf{b}_{L\times 1})$, where $\mathbf{x}_i$ is the input vector, $\mathbf{t}_i$ is the

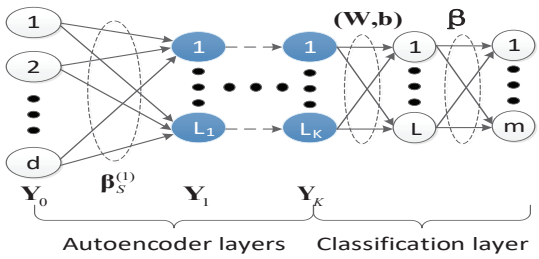

Fig. 1. The architecture of H-ELM.

desired output and L is the number of hidden neurons, the aim of ELM is to minimize the training error as

$$\min_{\boldsymbol{\beta}} \|\mathbf{H}\boldsymbol{\beta} - \mathbf{T}\|_2^2, \tag{1}$$

where $\mathbf{H}_{N\times L} = \mathrm{g}(\mathbf{W}\mathbf{X} + \mathbf{b}\cdot\mathbf{1}^T)^T$ is the hidden-layer output matrix, $\mathbf{1}$ is an all-one vector of dimension N, $\mathbf{X} = [\mathbf{x}_1\ \mathbf{x}_2\ \cdots\ \mathbf{x}_N]$ is the input matrix, $\mathbf{T} = [\mathbf{t}_1\ \cdots\ \mathbf{t}_N]^T$ is the desired output matrix, and $\mathrm{g}(\bullet)$ is the activation function. Then, the output weight $\boldsymbol{\beta}_{L\times M}$ is calculated by

$$\boldsymbol{\beta} = \mathbf{H}^{\dagger}\mathbf{T}, \tag{2}$$

where $\mathbf{H}^{\dagger}$ is the Moore-Penrose generalized inverse of $\mathbf{H}$.

Different from the conventional ELM, ELM-AE uses the input $\mathbf{X}$ as the desired output and the hidden neurons are learnt to represent the input data. In addition, the ℓ_2-norm regularized ELM-AE with orthogonal hidden node parameters are utilized in [1]. The network is trained by the minimization problem

$$\begin{array}{l}\min \frac{1}{2}C\left\|\mathbf{H}\boldsymbol{\beta}_A - \mathbf{X}^T\right\|_2^2 + \frac{1}{2}\left\|\boldsymbol{\beta}_A\right\|_2^2, \\ \text{s.t.}\mathbf{W}^T\mathbf{W} = \mathbf{I}, \mathbf{b}^T\mathbf{b} = \mathbf{I}.\end{array} \tag{3}$$

where C is the regularization parameter. The encoded output can be obtained by

$$\mathbf{Y} = \mathrm{g}\left(\boldsymbol{\beta}_A\mathbf{X}\right), \tag{4}$$

To address the issue of dense feature in ML-ELM [1], Tang *et al.* [2] proposed the ELM-SAE by imposing the ℓ_1-norm constraint on the output weight $\boldsymbol{\beta}_S$ to achieve the sparse encoded features as

$$\min \left\|\mathbf{H}\boldsymbol{\beta}_S - \mathbf{X}^T\right\|_2^2 + \|\boldsymbol{\beta}_S\|_1. \tag{5}$$

A fast iterative shrinkage-thresholding algorithm (FISTA) has been adopted to solve $\boldsymbol{\beta}_S$.

2.2. H-ELM

H-ELM consists of two independent components: a feature extraction with stacked AEs and a classification layers with ELM, as shown in Fig. 1. Assume K SAE layers are used and $\mathbf{Y}^{(k-1)}$ is the output of $(k-1)$-th layer with $\mathbf{Y}^{(0)} = \mathbf{X}$, the output $\mathbf{Y}^{(k)}$ of k-th layer is

$$\mathbf{Y}^{(k)} = \mathrm{g}\left(\boldsymbol{\beta}_S^{(k)}\mathbf{Y}^{(k-1)}\right), k = 1, \cdots, K, \tag{6}$$

where $\boldsymbol{\beta}_S^{(k)}$ is the output weight of the k-th SAE. The supervised ELM classifier in the last layer is trained as

$$\min \left\|\mathrm{g}(\mathbf{W}\mathbf{Y}^{(K)} + \mathbf{b}\cdot\mathbf{1}^T)^T\boldsymbol{\beta} - \mathbf{T}\right\|_2^2, \tag{7}$$

where $\mathbf{W}$ and $\mathbf{b}$ are the randomly generated weight and bias, and $\boldsymbol{\beta}$ is the output weight matrix.

3. THE PROPOSED EH-ELM

The ℓ_1-norm based SAE may suffer the overfitting problem and it takes long training time for high-dimensional and large data. To alleviate these deficiencies, a novel SMA is first proposed to obtain the sparse features with analytical solution. Then, the EH-ELM is developed by employing the H-ELM learning framework with the ℓ_1-norm based SAE replaced by the proposed SMA.

3.1. SMA

In [17], a random matrix based projection is developed based on the Johnson-Lindenstrauss (JL) lemma, which states that after projection, the distance of any pair of two vectors can be preserved within an arbitrarily small tolerance. Random projection has been widely used for dimension reduction and the random sparse matrix $\mathbf{R}$ can be generated with a simple distribution as [17]

$$r_{ij} = \begin{cases} +\sqrt{3} & \text{with probability } 1/6 \\ 0 & \text{with probability } 2/3 \\ -\sqrt{3} & \text{with probability } 1/6 \end{cases} \tag{8}$$

where r_{ij} denotes the element in the random sparse matrix $\mathbf{R}$. The random projection based on (8) is also known as the sparse JL transform. The random sparse matrix not only preserves the pairwise distance, but also introduces sparsity. To further exploit the advantage of ELM on using random hidden parameters, we propose two new schemes for generation of the weight matrix $\mathbf{W} = [w_{ij}]$ as follows:

$$\text{Scheme 1 } w_{ij} = \begin{cases} 0 & \text{with probability } 2/3, \\ \mathrm{U}(-1,1) & \text{with probability } 1/3, \end{cases} \tag{9}$$

$$\text{Scheme 2 } w_{ij} = \begin{cases} 0 & \text{with probability } 2/3, \\ \mathrm{N}(-1,1) & \text{with probability } 1/3, \end{cases} \tag{10}$$

where $\mathrm{U}(\cdot)$ and $\mathrm{N}(\cdot)$ are the Uniform and Gaussian distributions, respectively.

By virtue of the above described sparse random weight matrix, we proposed a random sparse matrix based AE (SMA) in this paper. The SMA generates the hidden-layer weight $\mathbf{W}_M$ and $\mathbf{b}_M$ according to (9) or (10) and solves the output-layer weight $\boldsymbol{\beta}_M$ by the following ℓ_2-regularized nonlinear ELM-AE

$$\min \frac{1}{2} C \left\| \mathbf{H}_{\mathrm{M}} \boldsymbol{\beta}_M - \mathbf{X}^T \right\|_2^2 + \frac{1}{2} \left\| \boldsymbol{\beta}_M \right\|_2^2 \tag{11}$$

with

$$\mathbf{H}_M = \mathrm{g}(\mathbf{W}_M \mathbf{X} + \mathbf{b}_M \mathbf{1}^T)^T, \tag{12}$$

where $\mathbf{1}$ is an all-one vector of dimension N. The solution to problem (11) can be obtained as

$$\begin{array}{l} \boldsymbol{\beta}_M = \mathbf{H}_M^T \left(\frac{\mathbf{I}}{C} + \mathbf{H}_M \mathbf{H}_M^T \right)^{-1} \mathbf{X}^T \quad \text{if} \quad N < L, \\ \boldsymbol{\beta}_M = \left(\frac{\mathbf{I}}{C} + \mathbf{H}_M^T \mathbf{H}_M \right)^{-1} \mathbf{H}_M^T \mathbf{X}^T \quad \text{if} \quad N \geq L. \end{array} \tag{13}$$

The encoded result can be derived as

$$\mathbf{Y} = \mathrm{g}\left(\boldsymbol{\beta}_M \mathbf{X} \right). \tag{14}$$

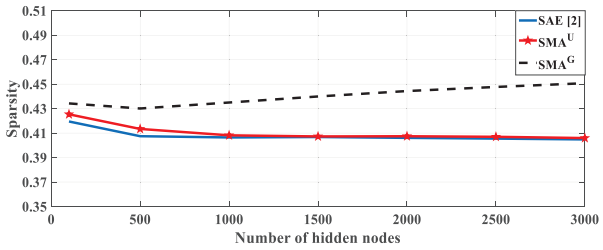

Fig. 2. Sparsity comparison among different sparse AEs.

To validate the effectiveness of the proposed SMA, an experiment is conducted to compare the sparsity of the proposed SMA and the existing ℓ_1-norm based SAE [2]. Both the Uniform distribution in (9) and the Gaussian distribution in (10) are tested to generate the random sparse matrix. The corresponding SMAs are denoted as SMA^U and SMA^G, respectively. The real-world NORB dataset [2] with 2048 features per sample is used. The criterion $m_s = (\sqrt{card(\boldsymbol{\beta})} - \|\boldsymbol{\beta}\|_1 / \|\boldsymbol{\beta}\|_2)(\sqrt{card(\boldsymbol{\beta})} - 1)^{-1}$ ($card(\boldsymbol{\beta})$ is the number of elements in $\boldsymbol{\beta}$) is employed for the sparsity evaluation of the output weight. Different numbers of hidden nodes of the AE ranging from 100 to 3000 are tested. For each number of hidden nodes, multiple trials are conducted and the average sparsity is calculated for comparisons. Fig. 2 shows the curves of sparsity obtained by the proposed SMA and the existing SAE [2]. It is evident that the proposed SMA with both random sparse matrix generating methods is effective in sparse encoding.

3.2. EH-ELM

By incorporating the H-ELM learning framework with the S-MA described above, an EH-ELM is developed. Algorithm 1 shows the pseudo codes of the EH-ELM. We first use the stacked SMAs to conduct the feature extraction. Then, the regularized ELM (RELM) is utilized for classification in the last layer. To achieve a good generalization performace, the orthogonal random input weights and bias are used as suggested in [1], i.e., $\mathbf{W}^T\mathbf{W}=\mathbf{I}$ and $\mathbf{b}^T\mathbf{b}=\mathbf{1}$. The output weight $\boldsymbol{\beta}$ in the last hidden layer is computed by

$$\begin{array}{l} \boldsymbol{\beta} = \mathbf{H}^T \left(\frac{\mathbf{I}}{C} + \mathbf{H}\mathbf{H}^T \right)^{-1} \mathbf{T} \quad \text{if} \quad N < L \\ \boldsymbol{\beta} = \left(\frac{\mathbf{I}}{C} + \mathbf{H}^T\mathbf{H} \right)^{-1} \mathbf{H}^T \mathbf{T} \quad \text{if} \quad N \geq L \end{array}, \tag{15}$$

where $\mathbf{T}$ is the desired output matrix of training data.

Algorithm 1 EH-ELM

Given:

Input data matrix $\mathbf{X}$, the regularization parameter and the number of hidden nodes $\left(C^{(k)}, L^{(k)}\right), k = 1, \cdots, K$, the activation function $\mathrm{g}(\bullet)$.

Training stage:

1. for $k = 1 : K$
   (a) Generate the random sparse input weight matrix $\mathbf{W}_M^{(k)}$ and bias $\mathbf{b}^{(k)}$ with (9) or (10)
   (b) Compute the hidden layer output matrix $\mathbf{H}_M^{(k)}$ with (12) and the output weight $\boldsymbol{\beta}_M^{(k)}$ with (13)
   (c) Derive the encoded result $\mathbf{Y}^{(k)}$ with (14)
2. Generate the orthogonal random matrix $\mathbf{W}$ and bias $\mathbf{b}$.
3. Compute the output weight $\boldsymbol{\beta}$ by (15).

Testing stage:

Given:

A testing input data $\mathbf{x}_p$

1. for $k = 1 : K$,
   $\mathbf{y}_p^{(k)} = \mathrm{g}\left(\boldsymbol{\beta}_M^{(k)} \mathbf{y}_p^{(k-1)} \right)$
2. Compute the output by
   $\mathbf{o}_p = \mathrm{g}\left(\mathbf{W}\mathbf{y}_p^{(K)} + \mathbf{b} \right)^T \boldsymbol{\beta}$

4. EXPERIMENTS AND COMPARISONS

Experiments and comparisons among ML-ELM [1], H-ELM[1] [2] and the proposed EH-ELM are given in this section. In the experiments, both the random sparse matrices generated by the Uniform and Gaussian distributions are tested in SMA. The corresponding EH-ELMs are denoted as EH-ELM^U and EH-ELM^G, respectively. Experiments are conducted on 12 high-dimensional and 11 low-dimensional benchmark classification datasets[2] specified in Tabel 1, as well as the MNIST and NORB datasets used in [2]. All algorithms are implemented with MATLAB codes on a computer with a 3.2GHz

[1]Codes available at http://www.ntu.edu.sg/home/egbhuang/elm codes.html

[2]http://www.ics.uci.edu/ mlearn/MLRepository.html

i5 CPU and 8GB RAM, except when applied to the MNIST and NORB datasets, the computer has a 3.4GHz i7 CPU and 32GB RAM.

For all three algorithms, three hidden layers are used, where ML-ELM consists of three AE layers and H-ELM and EH-ELM consist of two AE layers and one classifier layer. The parameter optimization is conducted by grid searches on $\{100, 500, 700, 1000, 5000, 12000, 15000\}$ for the hidden-layer node numbers and on $\{10^{-5}, 10^{-3}, 10^{3}, 10^{5}\}$ for the regularization factors. Average results on 10 independent trials are reported.

Table 1. Specifications of benchmark datasets.

Dataset	Train Data	Test Data	Features
ALLAML	43	29	7129
arcene	120	80	10000
Carcinom	105	69	9182
COIL20	860	580	1024
gisette	4200	2800	5000
GLIOMA	29	21	4434
HistALL	3326	830	4600
ORL64	240	160	4096
PCMAC	1166	777	3289
TOX171	102	69	5748
Yale64	105	60	4096
YaleB32	1436	978	1024
BreastTissue	75	31	9
bupa	200	145	6
mfeatall	1200	800	649
Cardiotocography	1701	425	21
diabetes2	576	192	8
randomfaces4ar	2100	500	540
Diabetic	806	345	19
randomAR	1800	800	540
magic	10000	9020	10
pcaAR	700	700	300
wine	100	78	13

4.1. Evaluation on high-dimensional datasets

Tables 2 shows the recognition rates and training time of ML-ELM, H-ELM and the proposed EH-ELM. As highlighted in boldface, EH-ELM obtains higher recognition rate with lower training time than ML-ELM and H-ELM on all 12 datasets. Since EH-ELM^U and EH-ELM^G cost close training time, we do not list the training time of EH-ELM^G in Table 2. It is noteworthy that, for ALLAML, arcene, GLIOMA, HistALL, YaleB32, EH-ELM^U offers more than 13.1%, 5%, 8.57%, 3.73%, 13.93% increments on the recognition rate over H-ELM. Meanwhile, it is also found that the EH-ELM^U performs slightly better than EH-ELM^G for most datasets.

4.2. Evaluation on low-dimensional datasets

Experiment results on the 11 low-dimensional datasets are shown in Table 3. As highlighted in Table 3, EH-ELM has better recognition performance than ML-ELM and H-ELM for all low-dimensional datasets. EH-ELM^U achieves a close accuracy to EH-ELM^G. In addition, the proposed EH-ELM learns faster than ML-ELM and H-ELM.

Table 2. Recognition rates (%) and training time (s) comparisons on high-dimensional datasets.

Dataset	ML-ELM [1]		H-ELM [2]		EH-ELM^U		EH-ELM^G
	Rate	Train time	Rate	Train time	Rate	Train time	Rate
ALLAML	89.66	21.33	83.45	16.45	**96.55**	**10.87**	96.55
arcene	82.5	21.46	80.5	18.80	**85.5**	**11.32**	85.5
Carcinom	91.3	21.40	94.78	18.18	**95.36**	**11.09**	95.07
COIL20	99.59	22.83	97.38	12.33	**99.66**	**11.29**	98.97
gisette	96.27	32.93	95.72	21.11	96.66	**19.49**	**96.89**
GLIOMA	62.86	21	62.86	14.27	**71.43**	**10.43**	66.67
HistALL	88.1	30.74	89.45	19.61	93.18	**17.41**	**93.25**
ORL64	89.88	21.10	96.88	14.09	**97.5**	**10.69**	96.75
PCMAC	74.13	23.79	83.55	14.79	**84.94**	**12.59**	84.07
TOX171	74.2	20.84	81.16	15.23	**82.61**	**10.68**	82.61
Yale64	78.33	20.70	87.33	13.98	**88.33**	**10.49**	88.33
YaleB32	93.66	24.03	84.11	12.99	98.04	**12.26**	**98.06**

Table 3. Recognition rates (%) and training time (s) comparisons on low-dimensional datasets.

Dataset	ML-ELM [1]		H-ELM [2]		EH-ELM^U		EH-ELM^G
	Rate	Train time	Rate	Train time	Rate	Train time	Rate
BreastTissue	61.29	20.22	51.61	10.59	**80.65**	**9.96**	74.19
bupa	66.90	0.005	65.38	0.003	71.59	**0.003**	**72.28**
mfeatall	98.65	23.62	98.07	12.47	**99.15**	**11.72**	99.1
Cardiotocography	84.33	24.75	88.85	12.60	**91.34**	**12.26**	90.31
diabetes2	64.06	0.05	69.48	0.02	**70.31**	**0.02**	67.4
randomfaces4ar	94.88	25.81	94.8	13.58	**98.92**	**13.04**	98.8
Diabetic	68.75	0.12	66.72	**0.06**	72.41	0.07	**73.16**
randomAR	78.35	25.33	84.45	13.31	89.5	**12.72**	**89.93**
magic	85.85	0.33	84.28	**0.19**	85.98	0.20	**86.17**
pcaAR	86.29	21.85	91.43	11.42	**92.49**	**10.92**	92.23
wine	97.69	20.31	97.44	10.53	**100**	**10.04**	98.72

4.3. Evaluation on MNIST and NORB datasets

Following [2], experiments on two complicated datasets, MNIST and NORB, are carried out in this section to vertify the superiority of EH-ELM. Table 4 shows the experimental results. It is obvious that EH-ELM wins the best recognition rate among all algorithms, and EH-ELM learns faster than ML-ELM and H-ELM. Especially, the training time by EH-ELM on the MNIST dataset is only about 30% and 80% of that by ML-ELM and H-ELM, respectively.

Table 4. Comparisons on MNIST and NORB datasets

Dataset	ML-ELM [1]		H-ELM [2]		EH-ELM^U		EH-ELM^G
	Rate	Train time	Rate	Train time	Rate	Train time	Rate
MNIST	99.00	281.71	98.99	101.39	**99.01**	78.23	99.00
NORB	88.27	251.59	90.65	165.72	**91.80**	150.34	91.77

5. CONCLUSIONS

Instead of using the ℓ_1-norm optimization based sparse AE, a novel random sparse matrix based AE (SMA) has been proposed in this paper. The proposed SMA is able to provide analytical solutions for the sparse feature encoding thereby learns faster than conventional AEs, which makes it more applicable for big data. Then, an enhanced hierarchical extreme learning machine algorithm (EH-ELM) has been developed by stacking the SMAS. Experimental results have been presented to verify the superiorities of the proposed EH-ELM algorithm.

References

[1] L. Chamara, H. Zhou, G.-B. Huang, and C.-M. Vong, "Representational learning with extreme learning machine for big data," *IEEE Intelligent Systems*, vol. 28, no. 6, pp. 31-34, Dec. 2013.

[2] J. Tang, C. Deng, and G.-B. Huang, "Extreme learning machine for multilayer perceptron," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, no. 4, pp. 809-821, Apr. 2016

[3] A. A. Mohammed, R. Minhas, Q. M. J. Wu, and M. A. Sid-Ahmed, "Human face recognition based on multidimensional PCA and extreme learning machine," *Pattern Recognition*, vol. 44, no. 10-11, pp. 2588-2597, 2011.

[4] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: Theory and applications," *Neurocomputing*, vol. 70, no. 1-3, pp. 489-501, Dec. 2006.

[5] J. Cao, K. Zhang, M. Luo, C. Yin, and X. Lai, "Extreme learining machine and adaptive sparse representation for image classification," *Neural Networks*, vol. 81, pp. 91-102, 2016.

[6] J. Cao, J. Hao, X. Lai, C.-M. Vong, and M. Luo, "Ensemble extreme learning machine and sparse representation classification algorithm," *Journal of the Franklin Institute*, vol. 353, no. 17, pp. 4526-4541, 2016.

[7] L. Zhang and D. Zhang, "Evolutionary Cost-Sensitive Extreme Learning Machine," *IEEE Transactions on Neural Networks and Learning Systems*, vol. PP, no. 99, pp. 1-16, 2016.

[8] J. Luo, C.-M. Vong, and P.-K. Wong, "Sparse Bayesian Extreme Learning Machine for Multi-classification," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 4, pp. 836-843, 2014.

[9] C.-M. Wong, C.-M. Vong, P.-K. Wong and J. Cao, "Kernel-Based Multilayer Extreme Learning Machines for Representation Learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. PP, no. 99, pp. 1-6, 2016.

[10] B. Chen, L. Xing, X. Wang, J. Qin and N. Zheng, "Robust Learning With Kernel Mean p-Power Error Loss," *IEEE Transactions on Cybernetics*, vol. PP, no. 99, pp. 1-13, 2017.

[11] B. Chen, S. Zhao, P. Zhu and J. Principe, "Quantized kernel recursive least squares algorithm," *IEEE Transactions on Neural Networks and Learning Systems*,?vol. 24, no. 9, 1484-1491, 2013.

[12] S. Nan, L. Sun, B. Chen, Z. Lin and K. Toh, "Density-dependent quantized least squares support vector machine for large data sets," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 1, pp. 94-106, 2017.

[13] P. Baldi, "Autoencoders, unsupervised learning, and deep architectures," *Journal of Machine Learning Research*, vol. 27, pp. 37-50, 2012

[14] R. Salakhutdinov and H. Larochelle, "Efficient Learning of Deep Boltzmann Machines," *Journal of Machine Learning Research*, vol. 9, pp. 693-700, 2010.

[15] W. Wang, Y. Huang, Y. Wang and L. Wang, "Generalized autoencoder: A neural network framework for dimensionality reduction," *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2014, pp. 496-503.

[16] G.-B. Huang, L. Chen, and C.-K. Siew, "Universal approximation using incremental constructive feedforward networks with random hidden nodes," *IEEE Transactions on Neural Networks*, vol. 17, no. 4, pp. 879-892, Jul. 2006.

[17] D. Achlioptas, "Database-friendly random projections," *20th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, 2001, pp. 274-281.

[18] N. Hurley and S. Rickard, "Comparing measures of sparsity," *IEEE Transactions on Information Theory*, vol. 55, no. 10, pp. 4723-4741, 2009