Dimension Reduction with Extreme Learning Machine

Liyanaraachchi Lekamalage Chamara Kasun, Yan Yang, Guang-Bin Huang and Zhengyou Zhang

Abstract—Data may often contain noise or irrelevant information which negatively affect the generalization capability of machine learning algorithms. The objective of dimension reduction algorithms such as Principal Component Analysis (PCA), Non-negative Matrix Factorization (NMF), random projection (RP) and auto-encoder (AE) is to reduce the noise or irrelevant information of the data. The features of PCA (eigenvectors) and linear AE is not able to represent data as parts (e.g. nose in a face image); On the other hand, NMF and non-linear AE is main by slow learning speed and RP only represents a subspace of original data. This paper introduces a dimension reduction framework which to some extend represents data as parts, has fast learning speed and learns the between-class scatter subspace. To this end, this paper investigates a linear and non-linear dimension reduction framework referred to as Extreme Learning Machine Auto-Encoder (ELM–AE) and Sparse Extreme Learning Machine Auto-Encoder (SELM–AE). In contrast to tied weight auto-encoder (TAE), the hidden neurons in ELM–AE and SELM–AE need not be tuned, their parameters (e.g. input weights in additive neurons) are initialized using orthogonal and sparse random weights respectively. Experimental results on USPS handwritten digit recognition dataset, CIFAR-10 object recognition and NORB object recognition data set show the efficacy of linear and non-linear ELM–AE and SELM–AE in terms of discriminative capability, sparsity, training time and Normalized Mean Square Error (NMSE).

Index Terms—Extreme Learning Machine (ELM), Principal Component Analysis (PCA), Non-negative Matrix Factorization (NMF), random projection (RP), auto-encoder (AE), Dimension reduction

I. INTRODUCTION

MACHINE learning algorithms aim to learn potential functions which map input data to target outputs. However, when input data consist of redundant features or noise, the generalization capability of machine learning methods will be negatively affected, and thus, feature extraction and dimension reduction such as Principal Component Analysis (PCA) [1], [2], Non-negative Matrix Factorization (NMF) [3], Random Projection (RP) [4], [5] and auto-encoder (AE) [6]–[18] are often required to preprocess training data in order to achieve better generalization performance.

L. L. C. Kasun and G.-B. Huang are with the School of Electrical and Electronic Engineering, Nanyang Technological University, Nanyang Avenue, Singapore 639798. E-Mail: (chamara001, egbhuang)@ntu.edu.sg

Y. Yang was with the Energy Research Institute @ NTU (ERI@N), Nanyang Technological University, Nanyang Avenue, Singapore 639798, is now with the Center of Intelligent Acoustics and Immersive Communications and School of Marine Science and Technology, Northwestern Polytechnical University, Xian 710072, China, and also with School of Information Science and Technology, Southeast University, Nanjing 210096, China. E-Mail: y.yang@nwpu.edu.cn

Z. Zhang is with the Microsoft Corporation, One Microsoft Way, Redmond WA 98052-6399, USA. E-Mail: zhang@microsoft.com

PCA is a holistic-based representation algorithm which represents the original data sample to different degrees, and learn a set of linearly uncorrelated features named principal components that describe the variance of data. Dimension reduction is achieved by projecting the input data via a subset of principal components that describes the most variance of the data. The features that have less contribution to the variances are considered to be less descriptive and therefore removed. PCA algorithm converges to a global minimum.

NMF is a parts-based representation algorithm which decomposes the data to a positive basis matrix and positive coefficient matrix. Positive basis matrix learns the commonly occurring parts in data (e.g. nose in a face image) and the positive coefficient matrix indicates the degree in which the commonly occurring parts reconstruct the data. NMF achieves dimension reduction by projecting the data along the positive basis matrix retaining the commonly occurring parts in the data while removing rarely occurring parts in the data1.

RP is a computationally efficient method for dimension reduction which retains the Euclidean distance between data points in the reduced dimension space. RP is achieved by projecting the data along an orthogonal random matrix [4] or sparse random matrix [5].

AE is a neural network based dimension reduction algorithm which has output data equal to input data. If the input layer neurons are larger than the hidden layer neurons AE perform dimension reduction. Linear AE learns the variance information similar to PCA [6], and non-linear AE learns non-linear features [7]–[10], [13]–[18]. Tied weight auto-encoder (TAE) is an AE which has the same weights for the input layer and output layer [13].

Extreme Learning Machine (ELM) [19]–[25] was originally proposed to train “generalized” single-hidden layer feedforward neural networks (SLFNs) with fast learning speed, good generalization capability and provides a unified learning paradigm for regression and classification [24]. Furthermore, ELM is efficient in hierarchal learning [26]–[28]. The essence of ELM is that the hidden node parameters can be assigned randomly independent of training data and need not be tuned, and the output weights are adjusted based on application dependent constraints. Its concrete biological evidence has been found recently [25], [29]–[33]. In contrast to the commonly used Back Propagation (BP) algorithm [34] which minimizes only the training error, ELM minimizes both the training error and the norm of the output weights. According to Bartlett’s

1For example, in a dataset which consists of humans the commonly occurring parts will be legs, hands, torso and faces while briefcase will be a rarely occurring part in the image.
theory [35], minimizing the norm of the weights leads to better generalization performance. ELM has been widely used in regression and classification problems due to its wide variety of feature mapping functions (sigmoid, hard-limit, Gaussian, multi-quadric, wavelet, Fourier series, etc) and its ability of handling both large and small datasets efficiently.

The outputs of ELM hidden nodes are referred to as the ELM feature mapping and can be used for dimension reduction (as shown in Figure 1). The commonly referred RP is a special case of ELM feature mapping [25]. In contrast to RP, ELM has a bias and non-linear activation function. Furthermore, ELM can approximate any continuous target function [19]–[25].

When the number of hidden layer neurons in an AE is smaller than that of input neurons it learns to reduce the dimension of data [36]. Motivated by the fact that AE could be used for dimension reduction, this paper investigates AE with random neurons in terms of: 1) theoretical investigation of the proposed linear AE; 2) Normalized Mean Square Error (NMSE); 3) discriminative capability of the features; 4) sparsity of the features and training time. This paper investigates linear and non-linear Extreme Learning Machine Auto-Encoder (ELM-AE) [36] and Sparse Extreme Learning Machine Auto-Encoder (SELM-AE) that has orthogonal and sparse random input neurons which are not tuned. The main contributions of the paper is as follows:

1) In contrast to the perception that linear AE learns the variance information, the proposed linear ELM-AE and linear SELM-AE learn the between-class scatter matrix which reduce the distance of data points belonging to the same cluster.
2) NMSE of linear and non-linear ELM-AE and SELM-AE is lower than that of the TAE.
3) Linear and non-linear ELM-AE and SELM-AE learn features which are robust to noise.
4) Features learned by ELM-AE (linear and non-linear) and SELM-AE (linear and non-linear) are at least discriminative as TAE, PCA and NMF.
5) ELM-AE (linear and non-linear) and SELM-AE (linear and non-linear) features are sparser than PCA, but not sparse as NMF; hence this may contain holistic-based and parts-based features.
6) Training time of linear and non-linear ELM-AE and SELM-AE is lower than TAE.

II. PRELIMINARIES

A. Extreme Learning Machines (ELM)

Feedforward neural networks are usually trained by Back-Propagation (BP) learning algorithm [34] in the past three decades. However BP faces bottlenecks such as slow learning speeds and local minimum problems. In contrast to the common understanding in neural networks community that all the hidden neurons in SLFNs need to be tuned, Extreme Learning Machine (ELM) theories [19]–[25] show that although the hidden nodes in SLFNs play critical roles they need not be tuned, instead they have universal approximation capability as long as these neurons are nonlinear piecewise continuous and randomly generated (e.g., random input weights and random biases in hidden nodes for additive neurons, or random centers and impact factors for Radial Basis Function neurons). This naturally overcomes the learning bottlenecks and barriers encountered by many conventional learning algorithms such as BP. Hence ELM has fast learning speed, and has both universal approximation and classification capabilities [19]–[24].

The network output of ELM network structure shown in Figure 1 is given by:

$$f_L(x) = \sum_{i=1}^{L} \beta_i h_i(x) = h(x)\beta$$  \hspace{1cm} (1)

where $\beta = [\beta_1, \ldots, \beta_L]^T$ is the output weight matrix between the hidden nodes and the output nodes while $h(x) = [h_1(x), \ldots, h_L(x)]$ is the hidden node output (random hidden feature) for the input $x \in \mathbb{R}^d$, and $h_i(x)$ is the output of the $i$-th hidden node. For example, for additive hidden node $h_i(x) = g(a_i, x, b_i)$, where $a_i$ and $b_i$ remain fixed after randomly generated. In ELM, the input data will be mapped to $L$ dimensional ELM random feature space $h(x)$. Given $N$ training samples $\{(x_i, t_i)\}_{i=1}^{N}$, ELM is to resolve the following learning problems:

**Minimize:** \[ ||\beta||_{p}^\sigma + C||H\beta - T||_{q}^{\sigma_2} \]  \hspace{1cm} (2)

where $\sigma_1 > 0$, $\sigma_2 > 0$, $p, q = 0, 1/2, 1, 2, \ldots, +\infty$, $T = [t_1, \ldots, t_N]^T$ consists of the targets and $H = [h(x_1), \ldots, h(x_N)]^T$. According to ELM learning theory, many types of feature mappings such as sigmoid, hard-limit, Gaussian, multi-quadric, Fourier series and wavelet, etc, can be adopted in ELM. When $C$ is infinitely large in Equation (2) the output weight $\beta$ is calculated as:

$$\beta = H^T T$$  \hspace{1cm} (3)

where $H^T$ is the Moore-Penrose generalized inverse [37], [38] of matrix $H$, which tends to achieve the smallest norm of $\beta$ while keeping the training error to the minimum.

Minimum norm of output weights based ELM achieves a better generalization performance and a more robust solution [24], [25]. The output weights $\beta$ from the ELM hidden layer to the output layer can be calculated as:

$$\beta = \left( \frac{1}{C} + H^T H \right)^{-1} H^T T$$  \hspace{1cm} (4)

or as:

$$\beta = H^T \left( \frac{1}{C} + HH^T \right)^{-1} T$$  \hspace{1cm} (5)

In detail, Huang et al. [24], [25] has shown that the equality constrained ELM given by Equation (4) and Equation (5) has fewer optimization constraints than Support Vector Machines (SVM) [39] and its variants including Least Square Support Vector Machine (LS-SVM) [40]. Unlike the SVM feature mapping function, ELM feature mapping $H$ is non-parametric. In contrast to SVM, ELM provides a unified learning algorithm for regression, binary-class classification and multi-class classification with higher generalization performance and much faster learning speed [24], [25].
The dimensions which describe the most variance of the data.

which describe the least variance of the data and preserving

via an orthogonal transformation so that the first dimension in

the projected space describes the most variance of the data

According to Equation (6), $V_1$ is maximized when $V_1$ is the first eigenvector of data $X_1$. Subsequent eigenvectors $V_j$ are calculated as:

$$V_j = \text{Maximize } \|V_j\| = 1 \left( \frac{V_j^T (X - \text{mean}(X))^T (X - \text{mean}(X)) V_1}{V_1^T V_1} \right)$$  (6)

According to Equation (6), $V_1$ is maximized when $V_1$ is the first eigenvector of data $X_1$. Subsequent eigenvectors $V_j$ are calculated as:

$$\hat{X}_{j-1}^T = X - \sum_{i=1}^{j-1} XV_{j-1}V_{j-1}^T$$

$$V_j = \text{Maximize } \|V_j\| = 1 \left( \frac{V_j^T \hat{X}_{j-1} \hat{X}_{j-1} V_j}{V_j^T V_j} \right)$$  (7)

Before performing PCA, the input data matrix $X$ should be standardized to have zero mean. The projected lower dimensional representation of the input data $X$ is created by multiplying the input data $X$ with the selected eigenvectors $V_j$ as:

$$X_{\text{proj}} = XV_j$$  (8)

C. Non-negative Matrix Factorization (NMF)

Non-negative Matrix Factorization (NMF) [3] performs parts-based non-linear dimensionality reduction. NMF factorizes a positive data matrix $X$ to a positive coefficient matrix $A$ and a positive basis matrix $W$. NMF factorization process is described as a square loss function as shown in Equation (9) or as a Kullback-Leibler divergence minimization function [41]. This paper adopts the following square loss which is commonly used in NMF:

$$\text{Minimize: } ||X - AW||^2$$

Subject to: $A \geq 0$, $W \geq 0$  (9)

The positive basis matrix $W = [w_1, \cdots, w_d]$ and positive coefficient matrix $A = [a_1, \cdots, a_d]$ are calculated using multiplicative update rules where $k$ is the number of iterations:

$$A^{k+1} = A^k \frac{XW^T}{A^k W^{k+1}(W^{k+1})^T}$$

$$W^{k+1} = W^k \frac{(A^k)^T X}{(A^k)^T A^k W^k}$$  (10)

In contrast to PCA, matrices $W$ and $A$ in NMF are not orthogonal, hence the captured features are not uncorrelated. There are two main concerns when NMF is used: 1) NMF converges to a local minimum [41]; 2) using NMF for dimension reduction is mathematically unjustifiable as it has not been proved that NMF learns statistical information of input data.

The positive basis matrix $W$ learns the most occurring parts in the data and the lower dimensional space $X_{\text{proj}}$ is created by multiplying the input data $X$ with the positive basis matrix $W$ as:

$$X_{\text{proj}} = XW^T$$  (11)

D. Random Projection (RP)

Random Projection (RP) aims to reduce the dimension of data $X$ while retaining the Euclidean information of data by projecting the data along an orthogonal [4] or sparse [5] random matrix $A$:

$$||x_i - x_j|| \approx ||x_i A - x_j A||$$  (12)
Orthogonal matrix $A$ is represented as:

$$A^T A = I$$  \hspace{1cm} (13)

Sparse random matrix [5] is represented as:

$$a_{ij} = 1/\sqrt{L} \begin{cases} +\sqrt{3} & p = 1/6 \\ 0 & p = 2/3 \\ -\sqrt{3} & p = 1/6 \end{cases}$$  \hspace{1cm} (14)

where $p$ represents the ratio of elements in the sparse random matrix $A$. Hence $\frac{2}{3}$ of the sparse random matrix values are zero, $\frac{1}{6}$ of the sparse random matrix values are $\sqrt{\frac{3}{L}}$, and $\frac{1}{6}$ of the sparse random matrix values are $-\sqrt{\frac{3}{L}}$. The time taken to generate sparse random weights are lower than orthogonal random weights [5] and the lower dimensional space $X_{proj}$ is calculated as:

$$X_{proj} = X A$$  \hspace{1cm} (15)

It is worth noting the difference between RP and ELM hidden layer feature mapping formed by random hidden neurons. Random projection uses linear projection which can only form a subspace of the original input space and does not have universal approximation capability. However, ELM uses non-linear hidden neurons which naturally has universal approximation and classification capabilities. The space formed by ELM feature mappings need not be subspace of the original input space, it can be a new space with non-linear relationship with the original input space. The new feature space formed by ELM feature mappings can have lower or higher dimensions than the original input space.

**E. Tied Weight Auto-Encoder (TAE)**

Similar to Restricted Boltzmann Machine (RBM) [42], Tied weight Auto-Encoder (TAE) [13] learns interesting features by using tied weights. TAE consists of an encoder stage and decoder stage as illustrated in Figure 2 and represents features of the input data in three different architectures: 1) compressed architecture; 2) sparse architecture; and 3) equal dimension architecture. (as shown in Figure 2).

**Compressed architecture**

The number of input neurons is larger than the number of neurons in the hidden layer. In this case, TAE learns to capture features from the hidden layer feature space, which has a lower dimensionality than the input data space.

**Sparse architecture**

The number of input neurons is smaller than the number of neurons in the hidden layer. In this case, TAE learns to capture features from the hidden layer feature space, which has a higher dimensionality than the input data space.

**Equal dimension architecture**

The number of input neurons is equal to the number of neurons in the hidden layer. In this case, TAE learns to capture features from the hidden layer feature space, which has the same dimensionality as the input data space.

As the main focus of this paper is dimension reduction compressed TAE architecture is discussed. The tied weight concept means: to use the same weights in the encoder stage and decoder stage of an Auto-Encoder (AE). This constrains the TAE to learn the same set of weights for dimension reduction and to reconstruct the data from the lower dimensional space.

**III. ELM-Feature Representation with Random Hidden Neurons**

Linear and non-linear Extreme Learning Machine Auto-Encoder (ELM-AE) and Sparse Extreme Learning Machine Auto-Encoder (SELM-AE) generates random hidden neurons (e.g., input weights and biases for additive neurons) in the encoder stage. Unlike TAE [42], the ELM-AE and SELM-AE do not use tied weights. As only the decoder stage weights are calculated in ELM-AE and SELM-AE, the computational cost is lower than TAE. Linear and non-linear ELM-AE and SELM-AE encoder stage weights are randomly generated based on random projection theories [4], [5] to retain Euclidean distance information. However, whether a non-linear activation function can be used in non-linear ELM-AE and SELM-AE is supported by ELM theories [19]–[24]:

![Fig. 2. TAE network architecture with tied weights $\beta_{TAE}$](image)
Lemma III.1. Universal approximation capability [19]–[24].

Given any bounded non-constant piecewise continuous function as the activation function in hidden neurons, if by tuning parameters of hidden neuron activation function SLFNs can approximate any target continuous function, then for any continuous target function \( f(x) \) and any randomly generated function sequence \( \{h_i(x)\}_{i=1}^L \), \( \lim_{L \to \infty} \| \sum_{i=1}^L \beta_i h_i(x) - f(x) \| = 0 \) holds with probability one with appropriate output weights \( \beta \).

Lemma III.1 shows that random hidden nodes (nonlinear mapping after random projections) can be used in different type of applications including clustering [43], [44], regression [23] and classification [24] by using different objective functions. For clustering the objective function is given by Equation (2).

Linear and non-linear ELM-AE and SELM-AE adopts an unsupervised learning as follows: input data are used as the output data: \( t=x \), input weights and biases of the random additive hidden nodes are chosen to be orthogonal for ELM-AE; sparse input weights and biases are chosen for the random hidden neurons in SELM-AE.

\[ X A \beta_{AE} - X \]

The orthogonal random hidden parameters of linear and non-linear ELM-AE architecture are calculated by Equation (20) accordingly with \( A = [a_1, \ldots, a_L] \) as the orthogonal random weight matrix between the input nodes and hidden nodes and \( b = [b_1, \ldots, b_L] \) is the bias vector of the hidden nodes. Vincent et al. [9] has shown that the hidden layer of an AE must retain information of the input data. Hence, orthogonal random parameters can be used in linear and non-linear ELM-AE to retain the Euclidean information of the input data as shown by Johnson-Lindenstrauss Lemma [4].

The output weights \( \beta_{AE} \) of linear and non-linear ELM-AE are responsible for the transformation from the feature space to input data \( (t=x) \) and satisfy:

\[ \min_{\beta_{AE}} ||H \beta_{AE} - X||^2 \]  

The analytical solution of Equation (21) is given by Equation (3) when \( T = X \).

The output weights \( \beta_{AE} \) of the compressed linear ELM-AE architecture representation with zero bias \( (b_i = 0) \) learn the between-class scatter matrix of data as shown in Theorem III.1. The derivation is motivated by linear AE [6]. In contrast, linear AE and PCA learn the variance information of data while NMF learn a parts-based representation of data.

Theorem III.1. Linear ELM-AE tries to minimize the objective function \( \min_{\beta_{AE}} ||X A \beta_{AE} - X||^2 \). When the number of input neurons in linear ELM-AE is larger than the number of hidden neurons \( (d > L) \), with zero bias \( (b_i = 0) \) and \( A^T A = I \), \( \beta_{AE} = A^T V V^T \), where \( V \) are the eigenvectors of covariance matrix \( X^T X \).

Proof. As shown in Figure 3 in linear ELM-AE we try to have \( \min_{\beta_{AE}} ||X A \beta_{AE} - X||^2 \), where \( \beta_{AE} = [\beta_{AE1}, \ldots, \beta_{AE L}]^T \) is the output weight matrix between the hidden nodes and the output nodes, \( A = [a_1, \ldots, a_L] \) are the orthogonal random weight and \( \{(x_i)_{i=1}^N\} \) are the input and output data. It is reasonable to assume that the input data matrix \( X \) has full rank in most applications.

The normal equations of the convex function \( E \) is given by:

\[ A^T X^T X A \beta_{AE} = A^T X^T X \]  

(22)

The \( d > L \) orthogonal random matrix is created by performing Singular Value Decomposition (SVD) [45] on a \( d \times d \) orthogonal random matrix where \( A_d g_d A_d^T = I \) and removing \( d-L \) singular values. Hence based on Eckart-Young-Mirsky Theorem [46], [47] \( A d \) is the best SVD low rank approximation of \( A_d A_d^T \) and \( AA^T \). However, note that \( A^T A = I \) for any \( d \) and \( L \) where \( d > L \).

\[ X^T X A \beta_{AE} = X^T X \]  

(23a)

\[ A = \beta_{AE} (\beta_{AE} \beta_{AE}^T)^{-1} \]  

(23b)

A. Extreme Learning Machine Auto-encoder (ELM-AE)

In contrast to the previous work of ELM-AE [36], this paper show that linear ELM-AE can learn the between-class scatter matrix, and also shows the efficacy of linear and non-linear ELM-AE in-terms of discriminative capability, sparsity, training time and Normalized Mean Square Error (NMSE). For compressed ELM-AE architecture, the hidden orthogonal random parameters project the input data to a lower dimension space and calculated by:

\[ h(x) = g(x A + b) = [h_1(x), \ldots, h_L(x)] \]

\[ = [g(a_1 x + b_1), \ldots, g(a_L x + b_L)] \]  

(20)
From Equation (23b) and $A^T A = I$, we have:

$$\beta_{AE} = A^T \beta_{AE} (\beta_{AE} A_{AE}^T)^{-1} \beta_{AE}$$  \hspace{1cm} (24)

In Equation (24) $P_{\beta_{AE}} = \beta_{AE} (\beta_{AE} A_{AE}^T)^{-1} \beta_{AE}$ is the orthogonal projection matrix of $\beta_{AE}$. Hence $\beta_{AE} = A^T P_{\beta_{AE}}$

The covariance of input data $X$ is $C_{XX} = \Sigma V V^T$, $V = [v_1 \cdots v_d]$ and $\Sigma = [\sigma_1 \cdots \sigma_d]$, where $V_i$ and $\sigma_i$ are the eigenvectors and eigenvalues of covariance matrix $X^T X$, respectively. The orthogonal projection $P_{\beta_{AE}}$ have a relationship with the orthogonal projection $P_{\beta_{AE}}$ as:

$$P_{\beta_{AE}} V = V^T \beta_{AE} (\beta_{AE} A_{AE}^T)^{-1} \beta_{AE} V$$  \hspace{1cm} (25)

By substituting $P_{\beta_{AE}} = \beta_{AE} (\beta_{AE} A_{AE}^T)^{-1} \beta_{AE}$ and $VV^T = I$ to Equation (25), we get the following relationship:

$$P_{\beta_{AE}} V = V^T P_{\beta_{AE}} V $$  \hspace{1cm} (26a)

$$P_{\beta_{AE}} = V P_{\beta_{AE}} V^T $$  \hspace{1cm} (26b)

Equation (23a) can be rearranged as:

$$X^T X \beta_{AE} = X^T X $$  \hspace{1cm} (27a)

$$C_{XX} \beta_{AE} = C_{XX} $$  \hspace{1cm} (27b)

$$\beta_{AE} A_{AE}^T C_{XX} A_{AE} \beta_{AE} = \beta_{AE} A_{AE}^T C_{XX} $$  \hspace{1cm} (27c)

$\beta_{AE}$ in Equation (27c) can be replaced with Equation (24) as:

$$P_{\beta_{AE}} A_{AE} A_{AE}^T C_{XX} A_{AE} \beta_{AE} = P_{\beta_{AE}} A_{AE} A_{AE}^T C_{XX} $$  \hspace{1cm} (28a)

$$(P_{\beta_{AE}} A_{AE} A_{AE}^T C_{XX} A_{AE} \beta_{AE})^T = C_{XX} A_{AE} A_{AE}^T P_{\beta_{AE}} $$  \hspace{1cm} (28b)

From Equation(28a) and Equation (28b), we get the following relationship:

$$P_{\beta_{AE}} A_{AE} A_{AE}^T C_{XX} A_{AE} P_{\beta_{AE}} = P_{\beta_{AE}} A_{AE} A_{AE}^T C_{XX} $$  \hspace{1cm} (29)

By using Equation (29) and Equation (26b), we can get the following relationship:

$$VP_{\beta_{AE}} V V^T A_{AE} V \Sigma V V^T = \Sigma V V^T A_{AE} V P_{\beta_{AE}} V V^T $$  \hspace{1cm} (30a)

$$P_{\beta_{AE}} V V^T A_{AE} V \Sigma = \Sigma V V^T A_{AE} V P_{\beta_{AE}} V $$  \hspace{1cm} (30b)

In Equation (30b) $P_{\beta_{AE}} V V^T A_{AE} V$ is diagonal matrix with rank $L$ with eigenvalues $1$ ($L$ times) and eigenvalue $0$ ($d-L$ times). Because $\Sigma$ is a diagonal with eigenvalues $\sigma_1 > \sigma_2 > \cdots > \sigma_d > 0$. Hence, $P_{\beta_{AE}} V V^T A_{AE} V = I_L$ with $L$ number of one’s (1) and $d-L$ zeros (0) in the diagonal.

$$P_{\beta_{AE}} V V^T A = I_L V V^T $$  \hspace{1cm} (31)

Equation (31) shows that $P_{\beta_{AE}} V = I_L$. By substituting $P_{\beta_{AE}} V = I_L$ and $\beta_{AE} = A^T P_{\beta_{AE}}$ to Equation (26b), the output weights $\beta_{AE}$ of linear ELM-AE is:

$$\beta_{AE} = A^T V V^T $$  \hspace{1cm} (32)

where $V$ is a $d \times L$ matrix.

This completes the proof.

Ding et al. [48] showed that between-class scatter matrix $MM^T$ can be represented by the eigenvector and its transpose $MM^T = V V^T$ (where $M = (m_1, \cdots, m_m)$ and $m_i$ is the mean of class $i$). Hence, learned features $\beta_{AE} = A^T V V^T$ is the between-class scatter matrix $VV^T$ projected along the orthogonal random weights $A$. Furthermore, Ding et al. [48] has shown that the between-class scatter reduces the distance of the data points belonging to the same cluster. Hence, linear ELM-AE removes dimensions that least effect the Euclidean distance of data points and is essentially the lowest variance dimensions. However in contrast to PCA, linear ELM-AE groups data points belonging to the same cluster.

Dimension reduction is achieved in linear and non-linear ELM-AE by projecting the data $X$ along the decoder stage weights $\beta_{AE}$ as:

$$X_{proj} = X \beta_{AE}^T $$  \hspace{1cm} (33)

B. Sparse Extreme Learning Machine Auto-Encoder (SELM-AE)

Sparsity is the main drive of creating a parts-based representation of an input data as shown in Sparse Coding [49], Reconstruction cost based Independent Component Analysis (RICA) [50] and NMF. Hence, for the proposed linear and non-linear SELM-AE, we impose sparsity by using sparse random parameters in the hidden layer. Furthermore, the hidden layer of an AE must retain information of the input data. To this end a sparse random matrix proposed by Achlioptas et al. [5] is used. Similar to orthogonal random matrix used in linear and non-linear ELM-AE, this sparse random matrix preserve the Euclidean distances between the data points in the projected space $X A$ while introducing sparsity. Hence, in linear and non-linear SELM-AE the random hidden node parameters are calculated by:

$$h(x) = g(x A + b) = [h_1(x), \cdots, h_L(x)]$$

$$= [g(a_1 \cdot x + b_1), \cdots, g(a_L \cdot x + b_L)] $$  \hspace{1cm} (34)

$$a_{ij} = b_i = 1/\sqrt{L} \begin{cases} +\sqrt{3} & p = 1/6 \\ 0 & p = 2/3 \\ -\sqrt{3} & p = 1/6 \end{cases}$$

where $A = [a_1, \cdots, a_L]$ is the sparse random matrix between the input nodes and the hidden nodes and $b = [b_1, \cdots, b_L]$ is the bias vector of the hidden nodes. The output weights $\beta_{SAE}$ of linear and non-linear SELM-AE are calculated as:

$$\text{Minimize: } ||H \beta_{SAE} - X||^2 $$  \hspace{1cm} (35)

Similar to ELM-AE, Theorem III.2 shows that output weights $\beta_{SAE}$ of linear SELM-AE learn the between-class scatter matrix.

Theorem III.2. Linear SELM-AE tries to minimize the function Minimize: $||X A \beta_{SAE} - X||^2$. When the number of hidden neurons in linear SELM-AE is smaller than the number of input neurons ($d > L$) and with zero bias ($b_i = 0$); $\beta_{SAE} = A^T V V^T$, where $V$ are the eigenvectors of covariance matrix $X^T X$.

Proof. We first show that $A^T A = I$ as sparse random matrix $A$ is not orthogonal.
A. Benchmark Datasets

TAE, PCA, NMF, ELM-AE (linear and non-linear) and SELM-AE (linear and non-linear) have been tested on USPS [51] handwritten digit recognition, CIFAR-10 object recognition [52] and NORB object recognition [53] datasets. These datasets were normalized to have zero mean and unit variance for TAE, PCA, ELM-AE (linear and non-linear) and SELM-AE (linear and non-linear), while for NMF data was normalized between 0 and 1. USPS dataset contains $16 \times 16$ dimensional digits from 0 to 9. There are 7291 training examples and 2007 testing data samples. CIFAR-10 dataset contains $3 \times 32 \times 32$ dimensional objects belonging to 10 classes. There are 40,000 training data samples, 10,000 samples for validation and 10,000 testing data samples. NORB data set contains $2 \times 96 \times 96$ dimensional object images of four legged animals, humans, airplanes, trucks and cars. There are 24,300 training data samples and 24,300 testing data samples. For our experiments we down sampled the NORB dataset to $2 \times 32 \times 32$.

B. Visual Investigation of Features Learned by ELM-AE and SELM-AE

Ding et al. [48] showed that the between-class scatter matrix $MM^T$ could be represented by eigenvectors $M = V V^T$.

IV. EXPERIMENTS

This paper shows that the proposed linear and non-linear ELM-AE and SELM-AE to an extend learns localized features and at least discriminative as TAE, PCA and NMF. Furthermore, the training time of linear and non-linear ELM-AE and SELM-AE is lower than TAE and NMF. Linear and non-linear ELM-AE and SELM-AE obtains lower NMSE between input and the reconstructed input data than TAE. To this end, this section compares the discriminative power, sparsity, training time and NMSE of the linear and non-linear ELM-AE, linear and non-linear SELM-AE, TAE, PCA and NMF features. The criteria for performance evaluations are as follows:

1) Discriminative capability measures the ability of the dimension reduction algorithm to learn important features and is measured by classifying the lower dimensional projections\(^2\) using ELM classifier. As it has been shown that SVM is suboptimal to ELM classifier [24], [25].

2) Sparsity describes the dimension reduction algorithm capability to learn localized features\(^3\) of the data and is measured by $L_2/L_1$ measure.

3) NMSE is calculated as:

$$\text{NMSE} = \frac{1}{N} \sum_{i=1}^{N} (x_i - \hat{x}_i)^2$$

where $\hat{x}_i$ is the largest value and smallest value of input data $x_i$, respectively.

The experiments were carried out in a high-end computer with a Xeon E5-2650 2 GHz processor and 256 GB RAM running MATLAB 2015b.

A. Benchmark Datasets

TAE, PCA, NMF, ELM-AE (linear and non-linear) and SELM-AE (linear and non-linear) have been tested on USPS [51] handwritten digit recognition, CIFAR-10 object recognition [52] and NORB object recognition [53] datasets. These datasets were normalized to have zero mean and unit variance for TAE, PCA, ELM-AE (linear and non-linear) and SELM-AE (linear and non-linear), while for NMF data was normalized between 0 and 1. USPS dataset contains $16 \times 16$ dimensional digits from 0 to 9. There are 7291 training examples and 2007 testing data samples. CIFAR-10 dataset contains $3 \times 32 \times 32$ dimensional objects belonging to 10 classes. There are 40,000 training data samples, 10,000 samples for validation and 10,000 testing data samples. NORB data set contains $2 \times 96 \times 96$ dimensional object images of four legged animals, humans, airplanes, trucks and cars. There are 24,300 training data samples and 24,300 testing data samples. For our experiments we down sampled the NORB dataset to $2 \times 32 \times 32$.

B. Visual Investigation of Features Learned by ELM-AE and SELM-AE

Ding et al. [48] showed that the between-class scatter matrix $MM^T$ could be represented by eigenvectors $M = V V^T$, \(^2\)See Equation (8), Equation (11), Equation (18), Equation (33) and Equation (37) to create lower dimension projections of PCA, NMF, TAE, ELM-AE (linear non-linear) and SELM-AE (linear and non-linear) respectively.

Bengio et al. [7] showed the advantages of a parts-based representation.
C. Measuring Sparsity of the Learned Features

This paper uses \( L_2/L_1 \) [55] sparsity measure to evaluate the sparsity of the learned features. Hence, \( L_2/L_1 \) measure indicates sparsity at an abstract level. \( L_2/L_1 \) measure is a ratio which weighs the features, so that large coefficients in the features get a small weight and small coefficients get a large weight. A higher \( L_2/L_1 \) measure indicates that there are few large feature coefficient values and more small feature coefficient values. A high \( L_2/L_1 \) value represents that the algorithm learns local information of data (parts-based features), a low value represents that the algorithm learn both local and global information of data.

D. Measuring Discriminative Power of the Learned Features

The discriminative power of the learned features of TAE, PCA, NMF, ELM-AE (linear and non-linear) and SELM-AE (linear and non-linear) were tested by reducing the dimensions of USPS, CIFAR-10 and NORB datasets (training and testing) using the following steps:

1) Calculate the \( \beta_{AE} \), \( \beta_{SAE} \), \( \beta_{TAE} \), \( \beta_{V} \) and \( \beta_{W} \) using ELM-AE (linear and non-linear), SELM-AE (linear and non-linear), TAE, PCA and NMF for \( k \) with training data.

2) Create lower dimensional training data and testing data for ELM-AE (linear and non-linear), SELM-AE (linear and non-linear), TAE, PCA and NMF projections by Equation (33), Equation (37), Equation (18), Equation (8) and Equation (11), respectively.

The average classification accuracy of thirty trials are reported for each ELM-AE (linear and non-linear), SELM-AE (linear and non-linear), TAE, PCA and NMF algorithm.

E. Performance Comparison of Different Algorithms in USPS, CIFAR-10 and NORB Datasets

Table II shows the average testing accuracy for features learned by ELM-AE (linear and non-linear), SELM-AE (linear and non-linear), TAE, PCA and NMF. The number of parameters to be selected are the number of features, the number of hidden neurons of the ELM classifier and the \( L_2 \) norm parameter of the ELM classifier for USPS, CIFAR-10 and NORB dataset. Table I shows the range of the parameters used to find the best accuracy of ELM-AE (linear and non-linear), SELM-AE (linear and non-linear), TAE, PCA and NMF.

Table II shows that non-linear SELM-AE lower dimensional projection allows ELM to classify 9 (0.44%) more samples than TAE projection, 8 (0.3%) more samples than PCA projection and 4 (0.2%) samples than NMF projection for USPS dataset. However, TAE projection require nearly 4 times more features than non-linear SELM-AE projection. For CIFAR-10 dataset non-linear SELM-AE lower dimensional projection allows ELM to classify 714 (7.14%) more samples than TAE projection, 60 (0.6%) more samples than PCA projection and 217 (2.17%) more samples than NMF projection. For NORB dataset non-linear ELM-AE lower dimensional projection allows ELM to classify 1458 (6%) more samples than TAE projection, 842 (3.51%) more samples than PCA projection and 1253 (5.22%) more samples than NMF projection.

The training time of linear and non-linear ELM-AE and SELM-AE low dimensional projection is at least 200 times faster than TAE and at least 8 times faster than NMF for USPS, CIFAR-10 and NORB datasets. However the training time of linear and non-linear ELM-AE and SELM-AE is in general equal to PCA in USPS and CIFAR-10 datasets but 5 times slower in NORB dataset.

NMSE values shows that linear and non-linear ELM-AE and SELM-AE is at least 4 times lower than TAE and similar to PCA for USPS, CIFAR-10 and NORB datasets. Hence, non-linear ELM-AE and SELM-AE learn a better model than TAE.

The corresponding sparsity values of ELM-AE, SELM-AE, TAE, PCA and NMF features for USPS, CIFAR-10 and NORB dataset is shown in Figure 5. Figure 5 shows that ELM-AE (linear and non-linear) and SELM-AE (linear and non-linear) are sparser than PCA for USPS, CIFAR-10 and NORB datasets, but not as sparse as NMF. Hence, linear and non-linear ELM-AE and SELM-AE learn local (parts-based) and global (holistic-based) information of data. Furthermore, Figure 5(b), Figure 5(d) and Figure 5(f) show that linear SELM-AE is sparser than linear ELM-AE, non-linear SELM-AE and non-linear ELM-AE for large features.

PCA algorithm learns global information (holistic-based) of the data, while NMF learns local information (parts-based) of data. In contrast TAE, linear and non-linear ELM-AE and SELM-AE learn both local and global information as they try to maximize the mutual information [9]. Hence they require more features than PCA and NMF.

V. Conclusions

Conventional AE tunes input weights and output weights iteratively to learn features of data. This paper investigates linear and non-linear ELM-AE and SELM-AE with orthogonal and sparse random hidden neurons. Orthogonal random and sparse random input neurons (e.g., random input weights and biases in additive neurons) are used to retain the Euclidean...
Fig. 4. The ELM-AE (linear and non-linear), SELM-AE (linear and non-linear), PCA and NMF lower dimensional projections of IRIS dataset.
(a) ELM-AE (linear and non-linear) and SELM-AE (linear and non-linear) is sparser than PCA for USPS dataset.

(b) Large features of linear SELM-AE is sparser than linear ELM-AE, non-linear SELM-AE and non-linear ELM-AE for USPS dataset.

(c) ELM-AE (linear and non-linear) and SELM-AE (linear and non-linear) is sparser than PCA for CIFAR-10 dataset.

(d) Large features of linear SELM-AE is sparser than linear ELM-AE, non-linear SELM-AE and non-linear ELM-AE for CIFAR-10 dataset.

(e) ELM-AE (linear and non-linear) and SELM-AE (linear and non-linear) is sparser than PCA for NORB dataset.

(f) Large features of linear SELM-AE is sparser than linear ELM-AE, non-linear SELM-AE and non-linear ELM-AE for NORB dataset.

Fig. 5. Sparsity values of ELM-AE (linear and non-linear), SELM-AE (linear and non-linear), TAE, PCA and NMF for USPS, CIFAR-10 and NORB datasets.
information of the data in the hidden layer [4], [5] as an AE must retain information of the data in the hidden layer [9]. As only the output weights must be calculated computational complexity of the proposed linear and non-linear ELM-AE and SELM-AE is lower.

In contrast to the common perception that linear AE learn the variance information of the data, the proposed linear ELM-AE and linear SELM-AE with random neurons in theory learn the between-class scatter matrix which reduces the distances of data points belonging to the same cluster.

NMSE values shows that non-linear ELM-AE and SELM-AE learn a better model than TAE.

Non-linear ELM-AE and SELM-AE hidden layer output is calculated in three steps: 1) use orthogonal or sparse random weights to project the data to a lower dimension projection which retains Euclidean information of the data; 2) add orthogonal or sparse random bias to the lower dimension projection which corrupts the Euclidean information of the data; 3) add a non-linear activation function to the output of the previous step which corrupts the output of previous step non-linearly. Hence, the decoding stage of non-linear ELM-AE and SELM-AE tries to learn features which are robust to noise by trying to map the output of the hidden layer to the data.

The experimental results show that the proposed ELM-AE (linear and non-linear) and SELM-AE (linear and non-linear) learn features more localized than PCA (higher sparsity than PCA), while not localized as NMF (lower sparsity than NMF). Furthermore, ELM-AE (non-linear and linear) and SELM-AE (linear and non-linear) features are at least discriminative as TAE, PCA and NMF, while PCA features are more discriminative than NMF. Hence this paper shows that ELM-AE (linear and non-linear) and SELM-AE (linear and non-linear) learn features which are discriminative and sparse.

REFERENCES
Yan Yang received her Ph.D. degree in Faculty of Engineering and the Environment, University of Southampton. She is a Senior Research Fellow in Energy Research Institute, Nanyang Technological University. Her research interests include machine learning, human machine interaction and system engineering.

Guang-Bin Huang (M’98-SM’04) received the B.Sc degree in applied mathematics and M.Eng degree in computer engineering from Northeastern University, P. R. China, in 1991 and 1994, respectively, and Ph.D degree in electrical engineering from Nanyang Technological University, Singapore in 1999. During undergraduate period, he also concurrently studied in Applied Mathematics department and Wireless Communication department of Northeastern University, P. R. China. He serves as an Associate Editor of Neurocomputing, Neural Networks, Cognitive Computation, and IEEE Transactions on Cybernetics. He is a senior member of IEEE. His current research interests include machine learning, computational intelligence, and extreme learning machines.

From June 1998 to May 2001, he worked as Research Fellow in Singapore Institute of Manufacturing Technology (formerly known as Gintic Institute of Manufacturing Technology) where he has led/implemented several key industrial projects (e.g., Chief designer and technical leader of Singapore Changi Airport Cargo Terminal Upgrading Project, etc). From May 2001, he has been working as an Assistant Professor and Associate Professor in the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore. He received Best Paper Award of IEEE Transactions on Neural Networks and Learning Systems in 2013.

Zhengyou Zhang is a principal researcher and research manager of the Multimedia, Interaction, and Communication (MIC) Group at Microsoft Research. His research interests include computer vision, speech signal processing, multisensory fusion, multimedia computing, real-time collaboration, and human-machine interaction. Zhang has PhD and DSc degrees in computer science from the University of Paris XI. He is a fellow of IEEE and the founding editor in chief of the IEEE Transactions on Autonomous Mental Development.