Neural Networks 77 (2016) 14–28

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

A Fast SVD-Hidden-nodes based Extreme Learning Machine for Large-Scale Data Analytics

Wan-Yu Deng [a,b,*], Zuo Bai [c], Guang-Bin Huang [c], Qing-Hua Zheng [d]

[a] School of Computer, Xian University of Posts & Telecommunications, Shaanxi, China
[b] School of Computer Engineering, Nanyang Technological University, Singapore
[c] School of Electrical & Electronic Engineering, Nanyang Technological University, Singapore
[d] Department of Computer Science and Technology, Xi'an Jiaotong University, China

ARTICLE INFO

Article history:
Received 4 February 2015
Received in revised form 3 August 2015
Accepted 7 September 2015
Available online 21 October 2015

Keywords:
Extreme Learning Machine
Singular value decomposition
Big data
Big dimensional data
Fast approximation method

ABSTRACT

Big dimensional data is a growing trend that is emerging in many real world contexts, extending from web mining, gene expression analysis, protein–protein interaction to high-frequency financial data. Nowadays, there is a growing consensus that the increasing dimensionality poses impeding effects on the performances of classifiers, which is termed as the "peaking phenomenon" in the field of machine intelligence. To address the issue, dimensionality reduction is commonly employed as a preprocessing step on the Big dimensional data before building the classifiers. In this paper, we propose an Extreme Learning Machine (ELM) approach for large-scale data analytic. In contrast to existing approaches, we embed hidden nodes that are designed using singular value decomposition (SVD) into the classical ELM. These SVD nodes in the hidden layer are shown to capture the underlying characteristics of the Big dimensional data well, exhibiting excellent generalization performances. The drawback of using SVD on the entire dataset, however, is the high computational complexity involved. To address this, a fast divide and conquer approximation scheme is introduced to maintain computational tractability on high volume data. The resultant algorithm proposed is labeled here as Fast Singular Value Decomposition-Hidden-nodes based Extreme Learning Machine or FSVD-H-ELM in short. In FSVD-H-ELM, instead of identifying the SVD hidden nodes directly from the entire dataset, SVD hidden nodes are derived from multiple random subsets of data sampled from the original dataset. Comprehensive experiments and comparisons are conducted to assess the FSVD-H-ELM against other *state-of-the-art* algorithms. The results obtained demonstrated the superior generalization performance and efficiency of the FSVD-H-ELM.

© 2016 Elsevier Ltd. All rights reserved.

1. Introduction

The notion of "Big Dimensionality" was established recently in Zhai, Ong, and Tsang (2014) to place emphasis on the growing phenomenon of problem dimensionality that is observed across diverse scenarios and applications in the real world. When addressing Big Data, volume often comes to mind naturally and take immediate precedence, since it presents the challenges pertaining to the scalability issue. Also, this is what directly comes to our mind when we refer to the term "Big". However, it is worth highlighting that researchers have largely considered the "Big Instance Size" factor of "Volume" in the Big Data analytics community, which refers to the massive amounts of data that we continue to produce daily. In the context of data analytics, volume can be defined as a product of *instance size* and *dimensionality* of the data. The issue of "Big Dimensionality", however, has received much lesser attention. For more coverage on this relatively under-explored topic of "Big Dimensionality" wherein the explosion of features (variables) brings about new challenges to computational intelligence and Big data, the reader is referred to Zhai et al. (2014). *Big dimensional* data is a growing trend that is emerging in many real world contexts, extending from web mining, gene expression analysis, protein–protein interaction to high-frequency financial data Today, there is a growing consensus that the increasing dimensionality poses impeding effects on the performances of classifiers, which is termed as the "peaking phenomenon" in the field of machine intelligence (Sarunas & Vitalijus, 1980). Previous methods (Hanchuan, Fulmi, & Chris, 2005) to address the high dimensionality characteristics of data have mainly concentrated

* Corresponding author at: School of Computer, Xian University of Posts & Telecommunications, Shaanxi, China.
E-mail address: wanyu.deng@gmail.com (W.-Y. Deng).

http://dx.doi.org/10.1016/j.neunet.2015.09.003
0893-6080/© 2016 Elsevier Ltd. All rights reserved.

on preprocessing the data with dimensionality reduction methods, keeping only the salient features and discarding the non-informative ones before building the classifier. In this paper, we present an attempt to address both *Big dimensionality* and *Big instance size* simultaneously in the context of Big Data analytic by embedding Singular Value Decomposition Hidden nodes into the classical Extreme Learning Machine.

In spite of the extensive works on ELM research, existing efforts still cannot arrive at representations and models that are elegant for Big data. Like all methods known to date, ELM suffers from the effects of 'curse of dimensionality and instance size'. The performance of ELM has been shown to deteriorate substantially and becomes unstable in the face of Big data. Taking this cue, this paper considers an implicit incorporation of dimension reduction mechanism into the classical ELM to achieve high generalization performances. In particular, we present a Fast Singular Value Decomposition-Hidden-nodes based Extreme Learning Machine (or FSVD-H-ELM in short) to cope with *Big dimensional and Big instance* data. We embed SVD hidden nodes into the classical Extreme Learning Machine to arrive at SVD-H-ELM and provide theoretical proofs on the optimality of using SVD hidden nodes over random hidden nodes. Further, instead of identifying the SVD hidden nodes directly from the Big dataset, a fast divide and conquer approximation scheme is also introduced to maintain computational tractability and scalability by deriving SVD hidden nodes from multiple random subsets of data sampled from the original Big dataset, in order to arrive at the Fast SVD-H-ELM or FSVD-H-ELM. Comprehensive experiments conducted on commonly used Big dataset indicate that the SVD nodes in the hidden layer are able to capture the underlying characteristics of the Big dataset well, leading to excellent generalization performances and efficiencies.

The rest of the paper is organized as follows: Section 2 gives a brief overview of the classical ELM. Section 3 details the proposed Fast Singular Value Decomposition-Hidden-nodes based Extreme Learning Machine (or FSVD-H-ELM). In Section 4, A discussion on the feature learning ability of the fast divide-and-conquer scheme is made. In Section 5, we summarize the experimental studies on twelve commonly used benchmark problems. Section 6 summarizes the main conclusions.

2. Review of Extreme Learning Machine and large-scale data analytics

2.1. Extreme Learning Machine

Feedforward neural networks play key roles in data analytic and have been widely applied in many applications for its promising generalization ability (Chaturvedi, Ong, & Arumugam, 2015; Feng, Ong, Lim, & Tsang, 2014; Schaefer, Krawczyk, Celebi, & Iyatomi, 2014; Seah, Ong, & Tsang, 2013; Seah, Tsang, & Ong, 2012, 2013). However, popular learning techniques face some challenging issues such as intensive human intervene and slow learning speed (Bullinaria & AlYahya, 2014). Although many effective algorithms, such as the back-propagation are available, training a neural network with all parameters adjustable is usually of high computational burden (Rummelhart, 1986; Salama, Hassanien, & Revett, 2013). To overcome such issue, a useful learning scheme, the Extreme Learning Machine (ELM), was suggested in Huang, Zhu, and Siew (2006) for single layer feedforward neural networks and subsequently extended to different variations such as local connected structure (Huang, Bai, Kasun, & Vong, 2015) and multi hidden layers structure (Kasun, Zhou, Huang, & Vong, 2013). ELM and its variations (Bueno-Crespo, García-Laencina, & Sancho-Gómez, 2013; Cambria et al., 2013; Chen, Peng, Zhou, Li, & Pan, 2014; Fernandez-Delgado, Cernadas, Barro, Ribeiro, & Neves, 2014; Huang, Chen, & Siew, 2006; Huang, Ding, & Zhou, 2010; Huang, Zhou, Ding, & Zhang, 2012; Rong, Ong, Tan, & Zhu, 2008) have been successfully used in the fields as object recognition (Huang et al., 2015), terrain-based navigation (Kan, Lim, Ong, Tan, & Yeo, 2013) activity recognition (Deng, Zheng, & Wang, 2014), time series (Butcher, Verstraeten, Schrauwen, Day, & Haycock, 2013), security assessment (Xu, Dong, Zhao, Zhang, & Wong, 2012), written character recognition (Chacko, Krishnan, Raju, & Anto, 2012), face recognition (Mohammed, Minhas, Wu, & Sid-Ahmed, 2011), gene selection and cancer classification (Saraswathi, Sundaram, Sundararajan, Zimmermann, & Nilsen-Hamilton, 2011). In essence ELM is a learning scheme whose hidden nodes need not be tuned and can be randomly generated. Then the ELM transforms the training of the neural network into a linear problem where the output weights of ELMs can be analytically determined instead of being tuned (Luo, Vong, & Wong, 2014). For the sake of efficiency, in real applications they may be determined in different ways such as with or without iterations, with or without incremental implementations, etc. Xu etc. (Lin, Liu, Fang, & Xu, 2015; Liu, Lin, Fang, & Xu, 2015) studied the theoretical feasibility of ELM and proved that via suitable activation functions, such as polynomials, NadarayaWatson and sigmoid functions, the ELMs can attain the theoretical generalization bound of the neural networks with all connections adjusted. Before ELM, the similar idea has been adopted earlier in Jaeger (2001); Jaeger and Haas (2004) as the echo state network method (ESN), in Maass, Natschläger, and Markram (2002) as the liquid state machine (LSM), in Lowe (1989) as RBF network with random centers, in Schmidt, Kraaijveld, and Duin (1992) as the feedforward neural networks with random weights and in Pao (1989) as the random vector functional-link network (RVFL). The relationships between ELMs and these earlier work please refer to the literatures (Huang, 2015; Huang, Huang, Song, & You, 2015).

The output function of ELM with L hidden nodes for generalized SLFNs is:

$$f_L(\mathbf{x}) = \sum_{i=1}^{L} \boldsymbol{\beta}_i h_i(\mathbf{x}) = \mathbf{h}(\mathbf{x})\boldsymbol{\beta} \tag{1}$$

where $\boldsymbol{\beta} = [\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_L]$ is the output weights between the hidden nodes to output nodes. $\mathbf{h}(\mathbf{x}) = [h_1(\mathbf{x}), \ldots, h_L(\mathbf{x})]$ is the output vector of the hidden layer with respect to the input $\mathbf{x}$; $h_i(\mathbf{x}) = g(\mathbf{w}_i, b_i, \mathbf{x})$, $\mathbf{w}_i \in \mathbf{R}^d$, $b_i \in R$ is the output of the ith *random hidden node* where the hidden node parameters (e.g. $\mathbf{w}$ and b) need not be tuned, in particular, can be *randomly* generated according to any continuous sampling distribution probability. $g(\mathbf{w}_i, b_i, \mathbf{x})$ is a nonlinear piecewise continuous function satisfying ELM universal approximation capability theorems (Huang, Chen et al., 2006). For example, such nonlinear piecewise continuous function can be but are not limited to:

(1) Sigmoid function

$$g(\mathbf{w}, b, \mathbf{x}) = \frac{1}{1 + \exp(-\lambda(\mathbf{w} \cdot \mathbf{x} + b))}. \tag{2}$$

(2) Fourier function

$$g(\mathbf{w}, b, \mathbf{x}) = \sin(\mathbf{w} \cdot \mathbf{x} + b). \tag{3}$$

(3) Hardlimit function

$$g(\mathbf{w}, b, \mathbf{x}) = \begin{cases} 1 & \text{if } \mathbf{w} \cdot \mathbf{x} - b \geq 0 \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

(4) Gaussian function

$$g(\mathbf{w}, b, \mathbf{x}) = \exp(-b\|\mathbf{x} - \mathbf{w}\|^2). \tag{5}$$

ELM theory (Huang, 2014; Huang, Chen et al., 2006; Huang et al., 2012; Huang, Zhu et al., 2006) shows that such $\mathbf{h}(\mathbf{x})$ has *universal approximation capability*, that is, $\lim_{L\to\infty} \|\mathbf{h}(\mathbf{x})\boldsymbol{\beta} - f(\mathbf{x})\|_F^2 = \lim_{L\to\infty} \| \sum_{i=1}^{L} \boldsymbol{\beta}_i h_i(\mathbf{x}) - f(\mathbf{x})\|_F^2 = 0$ holds with a probability one for appropriate output weights $\boldsymbol{\beta}$. Given training samples $\boldsymbol{\Omega} = \{(\mathbf{x}_i, \mathbf{t}_i)\}_{i=1}^N$, ELM theory aims to reach the smallest norm $\min_{\boldsymbol{\beta}} \|\boldsymbol{\beta}\|_F^2$ of output weights and training error $\min_{\boldsymbol{\beta}} \|\mathbf{H}\boldsymbol{\beta} - \mathbf{T}\|_F^2$ where $\mathbf{H}$ is the hidden layer output matrix (*randomized matrix*):

$$\mathbf{H} = \begin{bmatrix} g(\mathbf{w}_1, b_1, \mathbf{x}_1) & \cdots & g(\mathbf{w}_L, b_L, \mathbf{x}_1) \\ \vdots & \ddots & \vdots \\ g(\mathbf{w}_1, b_1, \mathbf{x}_N) & \cdots & g(\mathbf{w}_L, b_L, \mathbf{x}_N) \end{bmatrix}$$

and

$$\mathbf{T} = \begin{bmatrix} \mathbf{t}_1 \\ \vdots \\ \mathbf{t}_N \end{bmatrix} = \begin{bmatrix} t_{11} & \cdots & t_{1m} \\ \vdots & \ddots & \vdots \\ t_{N1} & \cdots & t_{Nm} \end{bmatrix}.$$

$\mathbf{h}(\mathbf{x})$ actually maps the data from the d-dimensional input space to the L-dimensional hidden layer random feature space. As a special case of $g(\mathbf{w}, b, \mathbf{x})$, $\mathbf{H}$ can be written as:

$$\mathbf{H} = g\left([\mathbf{X}, \mathbf{1}]\begin{bmatrix} \mathbf{W} \\ \mathbf{b} \end{bmatrix}\right) = \begin{bmatrix} g(\mathbf{w}_1 \cdot \mathbf{x}_1 + b_1) & \cdots & g(\mathbf{w}_L \cdot \mathbf{x}_1 + b_L) \\ \vdots & \ddots & \vdots \\ g(\mathbf{w}_1 \cdot \mathbf{x}_N + b_1) & \cdots & g(\mathbf{w}_L \cdot \mathbf{x}_N + b_L) \end{bmatrix}.$$

Numerous methods are now available for attaining the weight vector $\boldsymbol{\beta}$, such as the *No-Prop* algorithm suggested by Widrow, Greenblatt, Kim, and Park (2013) and the iterative algorithm developed in Bai, Huang, Wang, Wang, and Westover (2014). A popular and effective solution is given as follows:

$$\boldsymbol{\beta} = \mathbf{H}^{\dagger}\mathbf{T} \tag{6}$$

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

Recently, the unified ELM was introduced in Huang (2014) and Huang et al. (2012), in which $\boldsymbol{\beta}$ is solved as a constrained optimization problem:

$$\text{Minimize}_{\boldsymbol{\beta}} : \|\mathbf{H}\boldsymbol{\beta} - \mathbf{T}\|_p^{\alpha_1} + \frac{1}{\gamma}\|\boldsymbol{\beta}\|_q^{\alpha_2} \tag{7}$$

where $\alpha_1 > 0, \alpha_2 > 0, p, q = 0, \frac{1}{2}, 1, 2, \ldots, F, +\infty$ and γ is control parameter for the tradeoff of structural risk and experimental risk. Numerous efficient methods can be used to calculate the output weights $\boldsymbol{\beta}$ including but not limited to orthogonal projection method, iterative methods (Bai et al., 2014), and eigenvalue decomposition method (Golub & Van Loan, 2012). When $p, q = F$ and $\alpha_1, \alpha = 2$, a popular and efficient closed-form solution (Huang et al., 2012) is:

$$\boldsymbol{\beta} = \begin{cases} \mathbf{H}^{\top}\left(\frac{\mathbf{I}}{\gamma} + \mathbf{H}\mathbf{H}^{\top}\right)^{-1}\mathbf{T} & N \geq L \\ \left(\frac{\mathbf{I}}{\gamma} + \mathbf{H}^{\top}\mathbf{H}\right)^{-1}\mathbf{H}^{\top}\mathbf{T} & N \leq L. \end{cases} \tag{8}$$

2.2. Large-scale data analytic

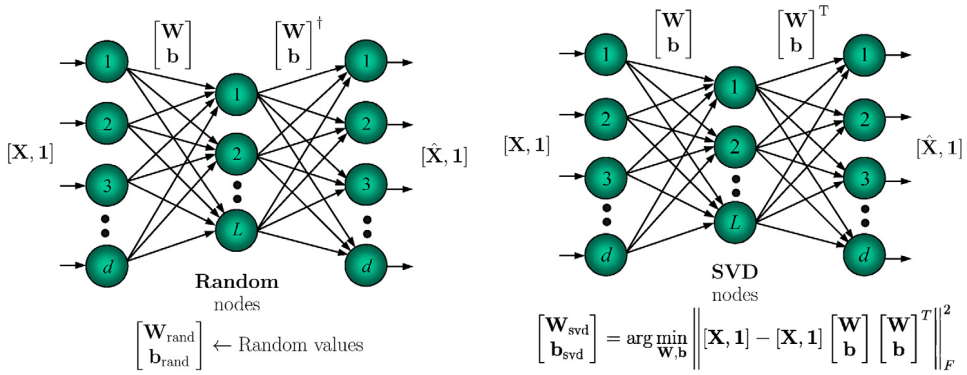
To process increasing amounts of data efficiently, many efficient and effective algorithms have been proposed. One kind of alternative methods directly avoids expensive computation burden of nonlinear decision functions, and builds large-scale *linear* algorithms on original feature space such as Bordes, Bottou, and Gallinari (2009), Chang, Hsieh, Chang, Ringgaard, and Lin (2010), Hsieh, Chang, Lin, Keerthi, and Sundararajan (2008), Joachims (2006), Shalev-Shwartz, Singer, Srebro, and Cotter (2011), Tsang, Kwok, and Cheung (2005), Yu, Hsieh, Chang, and Lin (2010), Zhang (2004) and Zhu, Chen, Wang, Zhu, and Chen (2009). This kind of algorithms has showcase competitive generalized performance at fast learning speed in some certain applications such as sparse high dimensional linear separated (or nearly) problems. On the other hand, arising number of researchers are devoting their time and efforts in variants of nonlinear kernel approximation (Wang, Djuric, Crammer, & Vucetic, 2011; Wang, Koby, & Slobodan, 2012; Zhang, Lan, Wang, & Moerchen, 2012) to overcome the challenge of large scale learning. For instances, AMM (Wang et al., 2011) captures non-linearity by assigning a number of linear hyperplanes to each of classes, and then learned via Stochastic Gradient Descent (SGD). LLSVM (Zhang et al., 2012) approximates kernel SVM optimization by a linear SVM using low-rank decomposition of the kernel matrix. BSGD-SVM (Wang et al., 2012) maintains a fixed number of support vectors in the model, and incrementally updates them during the SGD training. Hsieh, Si, and Dhillon (2014) proposed a divide-and-conquer solver for kernel SVMs (DC-SVM) where the kernel SVM problem is partitioned into smaller subproblems by clustering the data, so that each subproblem can be solved independently. Random Kitchen Sinks algorithm (Rahimi & Recht, 2009) approximates kernel expansions by means of multiplying the input with a Gaussian random matrix, followed by the application of a nonlinearity. From a survey of the literature (Zhai et al., 2014), the core challenges of Big Data have been widely established and can be summarized under the popular 5Vs, namely, Volume, Velocity, Variety, Veracity and Value. The present work takes special interests on the volume aspect of Big data by focusing on the massive amounts of data that have been generated across a wide range of sources. In particular, within the context of Big data analytics, volume can be defined by the product of instance size and dimensionality of the data. Taking this cue, in the rest of this section, we present the essential ingredients of our proposed Singular Value Decomposition-Hidden-nodes based Extreme Learning Machine, which is designed to work with *Big dimensional and Big instance* data.

3. Proposed method

3.1. Problem of interests

Given training set $\boldsymbol{\Omega} = \{(\mathbf{x}_i, \mathbf{t}_i) | \mathbf{x}_i \in \mathbf{R}^d, \mathbf{t}_i \in \mathbb{R}^m\}_{i=1}^N$, where d is the data dimension, N is instance size and m is the number of outputs. In a single-output regression problem, $\mathbf{t}_i \in R$ is a continuous real value ($\mathbb{R} = R$). For a m-class classification problem, $\mathbf{t}_i \in \{0, 1\}^m$ is a m-dimensional Boolean vector ($\mathbb{R} = \{0, 1\}$). If the original class label is p, the expected output vector of the m outputs is $\mathbf{t}_i = [0, \ldots, 0, \overset{p}{1}, 0, \ldots, 0]_{1\times m}$. In this case, only the pth element of $\mathbf{t}_i = [t_1, t_2, \ldots, t_m]$ is '1', while the rest of the elements are set to '0'. Binary classification is thus considered a special case of the multi-class problem where $m = 2$.

In the current work, our core interest is in the case where N and d are '*Big*'. On *Big instance* data, where only N is big, many effective algorithms has been proposed and studied (Krizhevsky, Sutskever, & Hinton, 2012; Mu, Huay, Fan, & Chang, 2014; Paisitkriangkrai, Shen, & van den Hengel, 2014; Shalev-Shwartz & Zhang, 2014; Verdaldi & Zisserman, 2012; Wang et al., 2011). However, despite the many ongoing efforts, majority (78.8%) (Zhai et al., 2014) of the studies on Big data analytic have been confined to *Big instance* data with features that are not so '*Big*', i.e., typically lower than 100,000 in dimensions. On *Big instance* and *Big dimensional* data, both N and d are very large, and to date less than 4% (Zhai et al., 2014)



(a) Random hidden nodes. (b) SVD hidden nodes.

Fig. 1. Random hidden nodes in classical ELM and SVD hidden nodes in our proposed method. For random hidden nodes, the parameters of the hidden nodes are *randomly* generated according to some continuous sampling distribution probability, while for SVD hidden nodes, the parameters of the hidden nodes are generated under the conditions of optimal low rank approximation.

of the studies in the computational intelligence community have considered real world data with features that range in the millions (Mao & Tsang, 2013; Tan, Tsang, & Wang, 2013). Thus, it is clear that the dimensionality of the algorithms under-studied is significantly lagging behind those being produced, mainly due to the many great challenges of *Big dimensional* and *Big instance* data.

3.2. Fast SVD-Hidden-nodes based Extreme Learning Machine for big dimensional and big instance data

By treating the hidden layer of ELM as unsupervised generative model (Jordan, 2002) with L hidden nodes, it plays the role of learning more useful (or discriminative) features from original data without losing much information. When the number of hidden nodes L is equal to or greater than the data dimensions, i.e., $L \geq d$, the input weights could be any unitary matrices to make reconstruction error approximating to zero, and thus it will become a trivial matrix; but by placing the constraint on the network by limiting the number of hidden nodes, we can learn essential embedding structure of the data. In special, the hidden layer as generative model is to re-represent the data with minimal reconstruction error $\left\|\begin{bmatrix}\mathbf{X} & \mathbf{1}\end{bmatrix} - \begin{bmatrix}\hat{\mathbf{X}} & \mathbf{1}\end{bmatrix}\right\|_F^2$ and minimal hidden nodes L, that is:

$$\text{Minimize}_{\mathbf{W},\mathbf{b}}: \left\|\begin{bmatrix}\mathbf{X} & \mathbf{1}\end{bmatrix} - \begin{bmatrix}\mathbf{X} & \mathbf{1}\end{bmatrix}\begin{bmatrix}\mathbf{W}\\ \mathbf{b}\end{bmatrix}\begin{bmatrix}\mathbf{W}\\ \mathbf{b}\end{bmatrix}^{\top}\right\|_F^2 \tag{9}$$

where $\mathbf{W} \in \mathbf{R}^{d\times L}$ and $\mathbf{b} \in \mathbf{R}^{1\times L}$ are input weights and bias, $\begin{bmatrix}\hat{\mathbf{X}} & \mathbf{1}\end{bmatrix}$ is the re-representation in L-dimension feature space of the original data.

In the classical ELM implementation, the above optimization condition is not considered and the hidden nodes have been randomly generated:

$$\begin{bmatrix}\mathbf{W}_{\text{rand}}\\ \mathbf{b}_{\text{rand}}\end{bmatrix} \leftarrow \text{Random values}. \tag{10}$$

The random hidden nodes architecture of the classical ELM is depicted in Fig. 1(a), which provides a simple approach for fast learning. However, when faced with *Big dimensional* data, the use of random hidden nodes tend to suffer since they fail to capture the true intrinsic characteristics of the *Big dimensionality*, resulting in a poor representation of the data in the hidden layer. Traditionally, researchers would use Principal Components Analysis (PCA) and other dimensionality reduction methods to work on the *Big dimensional* data as a form of pre-processing before building the classifiers (Fan, Richard, & Wu, 2009; Günter, Schraudolph, & Vishwanathan, 2007; Hanchuan et al., 2005; Jain, Duin, & Mao, 2000; Pierre, 1994). However, such a scheme would require additional memory space for archiving the preprocessed data (Bingham & Mannila, 2001). This can pose as an issue when dealing with Big data. In contrast to previous studies in the literature, we embed hidden nodes that are designed using Singular Value Decomposition (SVD) directly into the classical ELM to arrive at the proposed SVD-H-ELM. Particularly, we generate the SVD hidden nodes based on the optimal rank-L SVD as shown in Fig. 1(b). As proved in the following, SVD hidden nodes can satisfy the optimization condition of Eq. (9).

We consider the SVD of $[\mathbf{X}, \mathbf{1}]$ as

$$[\mathbf{X}, \mathbf{1}] \overset{\text{SVD}}{=} \mathbf{U}\mathbf{S}\mathbf{V}^{\top} \tag{11}$$

where $\mathbf{U} \in \mathbf{R}^{N\times N}$ and $\mathbf{V} \in \mathbf{R}^{d\times d}$ are unitary matrices, and $\mathbf{S} \in \mathbf{R}^{N\times d}$ is diagonal matrix with non-negative real numbers with decent order $\sigma_1 > \sigma_2 >, \ldots, > \sigma_N$ on the diagonal.

The optimal rank-L approximation of $[\mathbf{X}, \mathbf{1}]$ is given by zeroing out the $\min\{N, d\} - L$ trailing values of $\mathbf{S}$, that is:

$$[\mathbf{X}, \mathbf{1}] \triangleq \mathbf{U}_L\mathbf{S}_L\mathbf{V}_L^{\top} \tag{12}$$

where $\mathbf{U}_L$ and $\mathbf{V}_L$ denote the first $L < \min\{N, d\}$ columns of $\mathbf{U}$ and $\mathbf{V}$, and $\mathbf{S}_L = \text{diag}(\sigma_1, \sigma_2, \ldots, \sigma_L)$.

Parameter of the hidden nodes can then be obtained as

$$\begin{bmatrix}\mathbf{W}_{\text{SVD}}\\ \mathbf{b}_{\text{SVD}}\end{bmatrix} \leftarrow \mathbf{V}_L. \tag{13}$$

By replacing the random parameter of the hidden nodes with $(\mathbf{W}_{\text{SVD}}, \mathbf{b}_{\text{SVD}})$ and incorporating them as part of the ELM network thus arrive at the proposed SVD-H-ELM network. Consequently, the latent properties of the data can be appropriately represented via the SVD hidden nodes of the SVD-H-ELM.

Particularly, $(\mathbf{W}_{\text{SVD}}, \mathbf{b}_{\text{SVD}})$ gives the minimal *reconstruction error* with *Frobenius norm* according to the following theorem:

Theorem 1. *The SVD nodes with parameter* $(\mathbf{W}_{\text{SVD}}, \mathbf{b}_{\text{SVD}})$ *can provide more accurate representation of input* $\mathbf{X}$ *in the measure of* Frobenius norm *than random nodes with* $(\mathbf{W}_{\text{rand}}, \mathbf{b}_{\text{rand}})$.

Proof. Suppose that we have a classical ELM, in which parameter of the hidden nodes $(\mathbf{W}_{\text{Rand}}, \mathbf{b}_{\text{Rand}})$ is *randomly* generated according to any continuous sampling distribution probability, the *Frobenius norm* reconstruction error of the original data can be derived as:

$$\xi_{\text{Rand}} = \left\|[\mathbf{X}, \mathbf{1}] - [\mathbf{X}, \mathbf{1}]\begin{bmatrix}\mathbf{W}_{\text{Rand}}\\ \mathbf{b}_{\text{Rand}}\end{bmatrix}\begin{bmatrix}\mathbf{W}_{\text{Rand}}\\ \mathbf{b}_{\text{Rand}}\end{bmatrix}^{\dagger}\right\|_F^2. \tag{14}$$

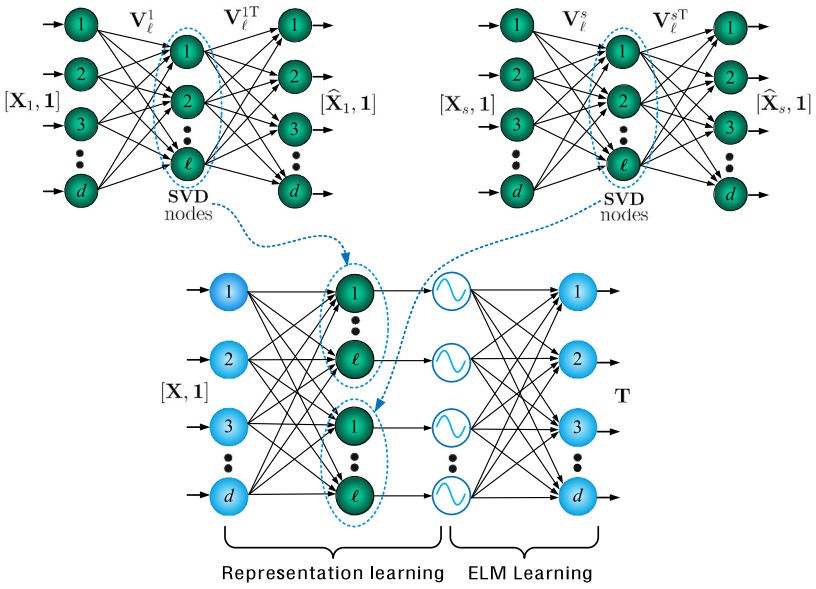


Fig. 2. Fast *divide-and-conquer* approximation scheme for Big data (*Big instance size* and *Big dimensionality*). In the *divide-and-conquer* scheme, the complete dataset is first sliced into multiple small subsets without overlapping according to a uniform distribution, and then each subset generates one part of the hidden nodes for FSVD-H-ELM such that reconstruction error is minimized independently.

For the SVD hidden nodes in SVD-H-ELM where the parameter of hidden nodes are generated as (13), the reconstruction error of the original data is given:

$$\xi_{\text{SVD}} = \left\| [\mathbf{X}, \mathbf{1}] - [\mathbf{X}, \mathbf{1}] \begin{bmatrix} \mathbf{W}_{\text{SVD}} \\ \mathbf{b}_{\text{SVD}} \end{bmatrix} \begin{bmatrix} \mathbf{W}_{\text{SVD}} \\ \mathbf{b}_{\text{SVD}} \end{bmatrix}^{\top} \right\|_F^2 = \|[\mathbf{X}, \mathbf{1}] - \mathbf{U}_L \mathbf{S}_L \mathbf{V}_L^T\|_F^2 \tag{15}$$

where $\mathbf{U}_L \mathbf{S}_L \mathbf{V}_L^T$ is optimal rank-L approximation of $[\mathbf{X}, \mathbf{1}]$ as shown in (12).

According to the Eckart–Young–Mirsky theorem (Carl & Gale, 1936), the rank-L approximation with the L largest singular values is the solution with the least reconstruction error, that is $\xi_{\text{SVD}} < \xi_{\text{Rand}}$ always holds. This means that SVD nodes yield more accurate representation of input $\mathbf{X}$ than random nodes in the hidden layer. □

Remark 1. In practice, noise often exists in the data. Assume that the noise is *i.i.d.* Gaussian noise. When dealing with data of low-to-medium dimensionality, the noise does not pose much of an issue (Günter et al., 2007). However, when dealing with data of *Big dimensionality*, the noise generally dominates in the last few principal components with smallest singular values, because the *i.i.d.* Gaussian noise is invariant across all principal components transformed by the input weight $[\mathbf{W}_{\text{SVD}}^{\top}, \mathbf{b}_{\text{SVD}}^{\top}]^{\top} = \mathbf{V}_L$, while the true signal variance is small in the last few components. In this case, the SVD hidden nodes in SVD-H-ELM provide more meaningful representations of the input $\mathbf{X}$ than random hidden nodes by disposing the last few components. The above thus summarizes the theoretical proof and analysis on the virtue of using SVD hidden nodes in SVD-H-ELM over the use of random hidden nodes in classic ELM.

3.3. A fast divide and conquer approximation scheme

With the availability of SVD hidden nodes, improved representations of the *Big dimensional* data can be achieved in the ELM over using random hidden nodes, thus enabling the proposed SVD-H-ELM to attain improved generalization performance and stability. The drawback of SVD-H-ELM, however, is the high computational complexity of SVD, especially when working with *Big instance* problems. Therefore, in this subsection, we present a fast approximation scheme as shown in Fig. 2, which takes inspiration from the concept of divide and conquer, to achieve computational tractability on large-scale problems. The proposed algorithm is then labeled here as the *Fast* Singular Value Decomposition-Hidden nodes-Extreme Learning Machine or FSVD-H-ELM. In FSVD-H-ELM, instead of identifying the SVD hidden nodes directly from the whole data, SVD hidden nodes are derived from multiple random subsets of data sampled from the original data, thus achieving improved scalability on *Big instance* and *Big dimensional* problems.

To date, many different methods to approximate the SVD at improved asymptotic running time have been described (Achlioptas & Mcsherry, 2007; Halko, Martinsson, & Tropp, 2011). For instance, in Halko et al. (2011), the decomposition was performed using QR to a random embedding matrix $\mathbf{C} = \mathbf{XM}$, in which $\mathbf{M} \in \mathbf{R}^{d \times k} (k < d)$ is a random matrix and $\mathbf{X} \in \mathbf{R}^{N \times d}$ is the data matrix. The total computation burden is around $\mathcal{O}(Ndk + N^2 k)$. In Achlioptas and Mcsherry (2007), sparsification was used on the data matrix by retaining an element x_{ij} with probability p_{ij} or replacing it with 0 otherwise. Although these approximation approaches can improve decomposition efficiency, since for *Big instances and Big dimension* data, the instances size N and the dimensions d are both very large, the approximate complexity is still considerable expensive. To tackle this problem, we adopt an efficient sampling scheme, where multiple subsets are constructed by picking random samples from the original data. The SVD hidden nodes derived from each of the multiple data subsets then serve as the approximation or generalization of the original data. Our scheme provides good approximation to the SVD of the original dataset with a bounded error. Intuitively, if the sampling is fair, these subsets can capture the general distribution of the data. In Section 4, theoretical proof and analysis were presented for our method. In summary, the SVD hidden nodes derived from these subsets can generally produce good representation of the *Big dimensional* data, while requiring tractable computational effort.

In our *divide-and-conquer* scheme, the complete data is first sliced (or divided) into $S = \lfloor \frac{N}{n} \rfloor$ subsets without overlapping such that each subset has around n samples. Each subset produces $\ell = \rho \cdot n$ hidden nodes where $0 < \rho < 1$ is the ratio of the retained rank, then s subsets $\{\mathbf{X}_r\}_{r=1}^{s}$ are randomly selected from S subsets

such that $L = \sum_{r=1}^{s} \ell$ where L is the total number of hidden nodes in FSVD-H-ELM.

For each data subset in $\{\mathbf{X}_r\}_{r=1}^{s}$, the optimal rank-$\ell$ approximation is obtained similar to (12):

$$[\mathbf{X}_r, \mathbf{1}] \triangleq \mathbf{U}_\ell^r \mathbf{S}_\ell^r \mathbf{V}_\ell^{rT}. \tag{16}$$

Simply assigning the parameter of ℓ hidden nodes, i.e. $\{(r-1)\cdot\ell+1, (r-1)\cdot\ell+2, \ldots, r\cdot\ell\}$th superscript nodes with $\mathbf{V}_\ell^r$ and concatenating them together, then the parameter of the L hidden nodes can be obtained as following

$$\begin{bmatrix} \mathbf{W}_{\text{FSVD}} \\ \mathbf{b}_{\text{FSVD}} \end{bmatrix} \Leftarrow \begin{bmatrix} \mathbf{W}_1 & \mathbf{W}_2 & \cdots & \mathbf{W}_s \\ \mathbf{b}_1 & \mathbf{b}_2 & & \mathbf{b}_s \end{bmatrix} \Leftarrow \left[V_\ell^1 \middle| V_\ell^2 \middle| \cdots \middle| V_\ell^s \right]. \tag{17}$$

Therefore, the hidden layer output matrix of SVD hidden nodes can be derived as:

$$\mathbb{P}_L = [\mathbf{X}, \mathbf{1}] \begin{bmatrix} \mathbf{W}_{\text{FSVD}} \\ \mathbf{b}_{\text{FSVD}} \end{bmatrix} = [\mathbf{X}, \mathbf{1}] \left[\mathbf{V}_\ell^1 \ \cdots \ \mathbf{V}_\ell^s \right] \tag{18}$$

$$\mathbf{H} = g(\mathbb{P}_L) = g\left(\mathbf{X} \left[\mathbf{V}_\ell^1 \ \cdots \ \mathbf{V}_\ell^s \right] \right). \tag{19}$$

3.4. Computational complexity

Here we analyze the computational complexities of the classical SVD (used in SVD-H-ELM) and our proposed divide-and-conquer SVD approximation scheme (used in FSVD-H-ELM). In particular, since the complexity of SVD on a $r \times c$ input data matrix is $\mathbf{X}\ \mathcal{O}\left(rc \cdot \min(r, c)\right)$, it can be easily derived that:

$$\mathrm{O}_{\text{classical-SVD}} = \mathcal{O}\left(Nd \cdot \min(N, d)\right) \tag{20}$$

and

$$\mathrm{O}_{\text{Divide and Conquer Approx-SVD}} = \mathcal{O}\left(\sum_{r=1}^{s} n^2 d \right). \tag{21}$$

Further, since the subset size n is typically set to be small ($n \ll N$) for the purpose of efficiency, it can be easily observed that the computational complexity of the fast *divide-and-conquer* approximation scheme $\mathrm{O}_{\text{Divide and Conquer Approx-SVD}}$ is much lower than that of the classical SVD $\mathrm{O}_{\text{classical-SVD}}$.

Remark 2. Since the SVD of each subset $\mathbf{X}_r$, $r = 1, 2, \ldots, s$ is conducted independently, parallel computing of the scheme can be conveniently realized. Therefore, the memory requirement is significantly reduced and the training speed can also be greatly improved.

Remark 3. The proposed algorithm is also suitable for online model update than other two-stage algorithms, where the latter involves dimensionality reduction as a pre-processing stage before the learning stage. For any two-stage algorithm, with the arrival of a new data point, besides the mapping matrix of dimensionality reduction that needs to be performed again in order to update, it also requires the learning algorithms to be recomputed or rebuilt. In the context of ELM, the number of hidden nodes L would need to be revised accordingly each time some new data points are received. On the contrary, with our proposed algorithm, L is the number of the retained components or the rank, i.e., $\sum_{r=1}^{s} \ell$. Thus, the value of L can be naturally adjusted by the rank-ℓ online updating in the online phase. Furthermore, with our scheme, there is no need for additional memory to store the preprocessed data after performing dimensionality reduction.

4. A discussion on the feature learning ability of the fast divide & conquer approximation scheme

In this section, we present an analysis on the *Frobenius reconstruction error* bound of the fast divide-and-conquer approximation scheme and compares it to the classical SVD on the original complete dataset.

Algorithm 1: Fast SVD-Hidden-Nodes Based Extreme Learning Machine

Input: Given training dataset $\Omega = \{(\mathbf{x}_i, \mathbf{t}_i)\}_{i=1}^{N}$, activation function g, regularized parameter γ, number of hidden nodes L, subset instance size n and hidden nodes ℓ that each subset generates.

Output: The model of ELM with Fast SVD hidden nodes: input layer weight W_{FSVD}, biases b_{FSVD}, and output layer weight $\boldsymbol{\beta}$.

1. Divide the training dataset into $\lfloor \frac{N}{n} \rfloor$ non-overlapping subsets, and then randomly select s subsets $\{\mathbf{X}_r\}_{r=1}^{s}$ such that $L = \sum_{r=1}^{s} \ell$;
2. **for** $r = 1$ *to* s **do**
3. Compute the optimal rank-ℓ approximation of $\mathbf{X}_r$: $X_\ell^r \triangleq \mathbf{U}_\ell^r \mathbf{S}_\ell^r \mathbf{V}_\ell^{r\top}$;
4. **end**
5. Compute parameter of the hidden nodes: $\begin{bmatrix} \mathbf{W}_{\text{FSVD}} \\ \mathbf{b}_{\text{FSVD}} \end{bmatrix} \Leftarrow \begin{bmatrix} \mathbf{W}_1 & \mathbf{W}_2 & \cdots & \mathbf{W}_s \\ \mathbf{b}_1 & \mathbf{b}_2 & & \mathbf{b}_s \end{bmatrix} \Leftarrow \left[V_\ell^1 \middle| V_\ell^2 \middle| \cdots \middle| V_\ell^s \right]$;
6. Compute the output matrix of hidden layer: $H = g\left([X, 1] \begin{bmatrix} \mathbf{W}_{\text{FSVD}} \\ \mathbf{b}_{\text{FSVD}} \end{bmatrix} \right)$;
7. Compute the output weights: $\boldsymbol{\beta} = H^\top \left(\frac{I}{\gamma} + HH^\top \right)^{-1} T$ or $\left(\frac{I}{\gamma} + H^\top H \right)^{-1} H^\top T$
8. **return** W_{FSVD}, b_{FSVD} *and* $\boldsymbol{\beta}$

First, we define the following notations which are used in the analysis:

- $\mathbb{P}_L = [\mathbf{X}, \mathbf{1}] \left[\mathbf{V}_\ell^1 \ \cdots \ \mathbf{V}_\ell^s \right]$: The L-dimensional representation of the original dataset (see (18)) in the *divide-and-conquer* scheme with s subsets $\{\mathbf{X}_r\}_{r=1}^{s}$;
- $\mathbb{P}_\ell^r = [\mathbf{X}, \mathbf{1}] \left[\mathbf{V}_\ell^r \right]$: The ℓ-dimensional representation of the original dataset in the *divide-and-conquer* scheme with single subset $\mathbf{X}_r$;
- $\mathbf{P}_\ell = [\mathbf{X}, \mathbf{1}] [\mathbf{V}_\ell]$: The ℓ-dimensional representation of the original dataset in the optimal classical rank-ℓ SVD;

Theorem 2. *Suppose N samples $\mathbf{X} = \{\mathbf{x}_i | \mathbf{x}_i \in \mathbf{R}^m\}_{i=1}^{N}$ are divided into $\lfloor \frac{N}{n} \rfloor$ subsets with subset size n, and each subset produces ℓ hidden nodes. The s subsets $\{\mathbf{X}_r\}_{r=1}^{s}$ are then randomly selected such that the $L = \sum_{r=1}^{s} \ell$ where L is the total number of hidden nodes in FSVD-H-ELM. Then with probability of at least $(1-\delta)$, the following inequality holds:*

$$\begin{aligned} &\|\mathbf{X}\mathbf{X}^\top - \mathbb{P}_L \mathbb{P}_L^\top\|_F \\ &\quad \le \|\mathbf{X}\mathbf{X}^\top - \mathbf{P}_\ell \mathbf{P}_\ell^\top\|_F + N\theta_{\text{D}} \left(\frac{64\ell}{n} \right)^{\frac{1}{4}} \\ &\qquad \times \left\{ 1 + \left[\frac{2(N - sn)\theta_{\text{dis}}^2 \log \frac{1}{\delta}}{s(N - \frac{1}{2})\left(1 - \frac{1}{2}\max\{sn, N - sn\}\right)} \right]^{\frac{1}{2}} \right\}^{\frac{1}{2}} \end{aligned}$$

where

$$\theta_{\text{D}} = \max_i \left(\mathbf{x}_i \mathbf{x}_i^\top \right)^{\frac{1}{2}} \quad \textit{and} \quad \theta_{\text{dis}} = \max_{i,j} (\mathbf{x}_i \mathbf{x}_i^\top + \mathbf{x}_j \mathbf{x}_j^\top - 2\mathbf{x}_i \mathbf{x}_j^\top)^{\frac{1}{2}}.$$

Proof. For $r \in [1, s]$, let $\mathbf{Z}_r = \sqrt{N/b}\mathbf{X}\mathcal{S}_r$, where $\mathcal{S}_r$ denotes the selection matrix corresponding to the subset $\mathbf{X}_r$. By definition of $\mathbb{P}_L$, the following condition holds:

$$\begin{aligned}\left\|\mathbf{X}\mathbf{X}^\top - \mathbb{P}_L\mathbb{P}_L^\top\right\|_F^2 &\le \frac{1}{s}\left\|\sum_{i=r}^{s}\mathbf{X}\mathbf{X}^\top - \mathbb{P}_\ell^r\mathbb{P}_\ell^{rT}\right\|_F^2 \\ &\le \frac{1}{s}\sum_{i=r}^{s}\left\|\mathbf{X}\mathbf{X}^\top - \mathbb{P}_\ell^r\mathbb{P}_\ell^{rT}\right\|_F^2. \end{aligned} \tag{22}$$

Note that $\mathbb{P}_\ell^r\mathbb{P}_\ell^{rT} \approx \left(\mathbf{X}\mathbf{X}_r^\top\right)(\mathbf{X}_r\mathbf{X}_r)^{-1}\left(\mathbf{X}_r\mathbf{X}^\top\right)$ is equivalent to the standard Nyström approximation of $\mathbf{X}\mathbf{X}^\top$. Hence, the following general inequality holds for the *Frobenius error* of the standard Nyström approximation method [5]:

$$\begin{aligned}\left\|\mathbf{X}\mathbf{X}^\top - \mathbb{P}_\ell^r\mathbb{P}_\ell^{rT}\right\|_F^2 &\le \left\|\mathbf{X}\mathbf{X}^\top - \mathbf{P}_\ell\mathbf{P}_\ell^\top\right\|_F^2 \\ &\quad + N\theta_{\mathrm{D}}\sqrt{64\ell}\left\|\mathbf{X}\mathbf{X}^\top - \mathbf{Z}_r\mathbf{Z}_r^\top\right\|_F \end{aligned} \tag{23}$$

where $\|\mathbf{X}\mathbf{X}^\top - \mathbf{P}_\ell\mathbf{P}_\ell^\top\|_F^2$ is the *Frobenius error* bound of the classical SVD with reduced rank-ℓ. Substituting (23) into (22), we have:

$$\begin{aligned}\|\mathbf{X}\mathbf{X}^\top - \mathbb{P}_L\mathbb{P}_L^\top\|_F^2 &\le \|\mathbf{X}\mathbf{X}^\top - \mathbf{P}_\ell\mathbf{P}_\ell^\top\|_F^2 \\ &\quad + \frac{N\theta_{\mathrm{D}}\sqrt{64\ell}}{s}\sum_{i=r}^{s}\|\mathbf{X}\mathbf{X}^\top - \mathbf{Z}_r\mathbf{Z}_r^\top\|_F. \end{aligned} \tag{24}$$

Denote

$$\phi(\mathcal{S}_r) = \frac{1}{s}\sum_{r=1}^{s}\|\mathbf{X}\mathbf{X}^\top - \mathbf{Z}_r\mathbf{Z}_r^\top\|_F. \tag{25}$$

Let $\mathcal{S}_r'$ be a sampling matrix that comprises the same columns as $\mathcal{S}_r$ except for one, and let $\mathbf{Z}_r'$ denote $\sqrt{N/b}\mathbf{X}\mathcal{S}_r'$. Changing one column of the full sample $\mathcal{S}$ (the selection matrix corresponding to $\{\mathbf{X}_r\}_{r=1}^s$) changes only one subsample $\mathcal{S}_r$ and thus only the single term $\|\mathbf{X}\mathbf{X}^\top - \mathbf{Z}_r\mathbf{Z}_r^\top\|_F$. Let $\mathbf{z}$ and $\mathbf{z}'$ denote the only differing column of $\mathbf{Z}_r$ and $\mathbf{Z}_r'$, then we have:

$$\begin{aligned}|\phi(\mathcal{S}_r') - \phi(\mathcal{S}_r)| &\le \frac{1}{s}\left\|\mathbf{z}'\mathbf{z}'^\top - \mathbf{z}\mathbf{z}^\top\right\|_F \\ &= \frac{1}{s}\left\|\mathbf{z}'\mathbf{z}'^\top - \mathbf{z}'^\top + \mathbf{z}'^\top - \mathbf{z}\mathbf{z}^\top\right\|_F \\ &= \frac{1}{s}\left\|(\mathbf{z}' - \mathbf{z})\mathbf{z}'^\top + \mathbf{z}(\mathbf{z}'^\top - \mathbf{z}^\top)\right\|_F \\ &\le \frac{2}{s}\left\|\mathbf{z}' - \mathbf{z}\right\|_F \max\{\|\mathbf{z}\|_F, \|\mathbf{z}'\|_F\}. \end{aligned} \tag{26}$$

Thus, the columns of $\mathbf{Z}_r$ are those of $\mathbf{X}$ as scaled by $\sqrt{N/n}$. The norm of the difference for two columns of $\mathbf{X}$ can be viewed as the norm of the difference for two feature vectors and thus can be derived to be bounded by θ_{dis}. Similarly, the norm of a single column of $\mathbf{X}$ is bounded by θ_{D}. This leads to the following inequality:

$$|\phi(\mathcal{S}_r') - \phi(\mathcal{S}_r)| \le \frac{2N\theta_{\mathrm{dis}}\theta_{\mathrm{D}}}{sn}. \tag{27}$$

Further, following *Corollary* 2 of Kumar, Mohri, and Talwalkar (2012), the expectation of $\phi(\mathcal{S}_r)$ can be bounded as follows:

$$E[\phi(\mathcal{S}_r)] = \frac{1}{s}\sum_{r=1}^{s}E\left[\|\mathbf{X}\mathbf{X}^\top - \mathbf{Z}_r\mathbf{Z}_r^\top\|_F\right] \le \frac{N\theta_{\mathrm{D}}}{\sqrt{n}}. \tag{28}$$

And following *Theorem* 1 of Kumar, Mohri, and Talwalkar (2009), we know:

$$\mathcal{P}\left[\phi(\mathcal{S}_r) - E\left[\phi(\mathcal{S}_r)\right] > \epsilon\right] \le \exp\left(\frac{-2\epsilon^2}{\alpha(sn, N)|\phi(\mathcal{S}_r) - \phi(\mathcal{S}_r')|^2}\right)$$

where

$$\alpha(sn, N) = \frac{sb(N - sn)}{\left(N - \frac{1}{2}\right)\left(1 - 1/\left(2\max\{sn, N - sn\}\right)\right)}. \tag{29}$$

Denote $\delta = \exp\left(\frac{-2\epsilon^2}{\alpha(sn,N)|\phi(\mathcal{S}_r) - \phi(\mathcal{S}_r')|^2}\right)$ then we have:

$$\log\delta = \frac{-2\epsilon^2}{\alpha(sn, N)|\phi(\mathcal{S}_r) - \phi(\mathcal{S}_r')|^2} \tag{30}$$

and then we have:

$$\epsilon = \left[\frac{\alpha(sn, N)}{2}|\phi(\mathcal{S}_r) - \phi(\mathcal{S}_r')|^2\log\frac{1}{\delta}\right]^{\frac{1}{2}}. \tag{31}$$

From the above, it is noted that with probability $Pr[\phi(\mathcal{S}_r) - E(\phi(\mathcal{S}_r)) \ge \epsilon] \le \delta$, the following condition holds:

$$\phi(\mathcal{S}_r) - E[\phi(\mathcal{S}_r)] \ge \left[\frac{\alpha(sn, N)}{2}|\phi(\mathcal{S}_r) - \phi(\mathcal{S}_r')|^2\log\frac{1}{\delta}\right]^{\frac{1}{2}} \tag{32}$$

then,

$$\begin{aligned}\phi(\mathcal{S}_r) &\ge \left[\frac{\alpha(sn, N)}{2}|\phi(\mathcal{S}_r) - \phi(\mathcal{S}_r')|^2\log\frac{1}{\delta}\right]^{\frac{1}{2}} + E[\phi(\mathcal{S}_r)] \\ &= \left[\frac{\alpha(sn, N)}{2}\left(\frac{2N\theta_{\mathrm{dis}}\theta_{\mathrm{D}}}{sn}\right)^2\log\frac{1}{\delta}\right]^{\frac{1}{2}} + \frac{N\theta_{\mathrm{D}}}{\sqrt{n}}. \end{aligned} \tag{33}$$

In other words, with at least $(1 - \delta)$ probability, the following condition holds:

$$\phi(\mathcal{S}_r) \le \left[\frac{\alpha(sn, N)}{2}\left(\frac{2N\theta_{\mathrm{dis}}\theta_{\mathrm{D}}}{sn}\right)^2\log\frac{1}{\delta}\right]^{\frac{1}{2}} + \frac{N\theta_{\mathrm{D}}}{\sqrt{n}}. \tag{34}$$

Combining (24), (25) and (34) we arrive at:

$$\begin{aligned}\|\mathbf{X}\mathbf{X}^\top - \mathbb{P}_L\mathbb{P}_L^\top\|_F^2 &\le \|\mathbf{X}\mathbf{X}^\top - \mathbf{P}_\ell\mathbf{P}_\ell^\top\|_F^2 \\ &\quad + N\theta_{\mathrm{D}}\sqrt{64\ell}\left\{\left[\frac{\alpha(sn, N)}{2}\left(\frac{2N\theta_{\mathrm{dis}}\theta_{\mathrm{D}}}{sn}\right)^2\log\frac{1}{\delta}\right]^{\frac{1}{2}} + \frac{N\theta_{\mathrm{D}}}{\sqrt{n}}\right\}. \end{aligned} \tag{35}$$

Or

$$\begin{aligned}\|\mathbf{X}\mathbf{X}^\top - \mathbb{P}_L\mathbb{P}_L^\top\|_F &\le \|\mathbf{X}\mathbf{X}^\top - \mathbf{P}_\ell\mathbf{P}_\ell^\top\|_F + N\theta_{\mathrm{D}}\left(\frac{64\ell}{n}\right)^{\frac{1}{4}} \\ &\quad \times\left\{1 + \left[\frac{2(N - sn)\theta_{\mathrm{dis}}^2\log\frac{1}{\delta}}{s\left(N - \frac{1}{2}\right)\left(1 - \frac{1}{2}\max\{sn, N - sn\}\right)}\right]^{\frac{1}{2}}\right\}^{\frac{1}{2}}. \end{aligned} \tag{36}$$

□

From the above, it is clear that the *divide-and-conquer* approximated scheme is capable of reconstructing the original input $\mathbf{X}$ with a bounded error. Further, it mitigates the 'overfitting' phenomena with features that tends to have improved generalized predictive performance since our current scheme resembles a form of ensemble of multiple SVD denoised data. Here we treat the random subsets as a result of rows-wise denoising using random Boolean values. It is worth noting that ensemble learning and denoising technology have been proven in the literature to learn more useful feature representation (Vincent, Larochelle, Lajoie, Bengio, & Manzagol, 2010). Our experimental results in the next section also verify the hypothesis.

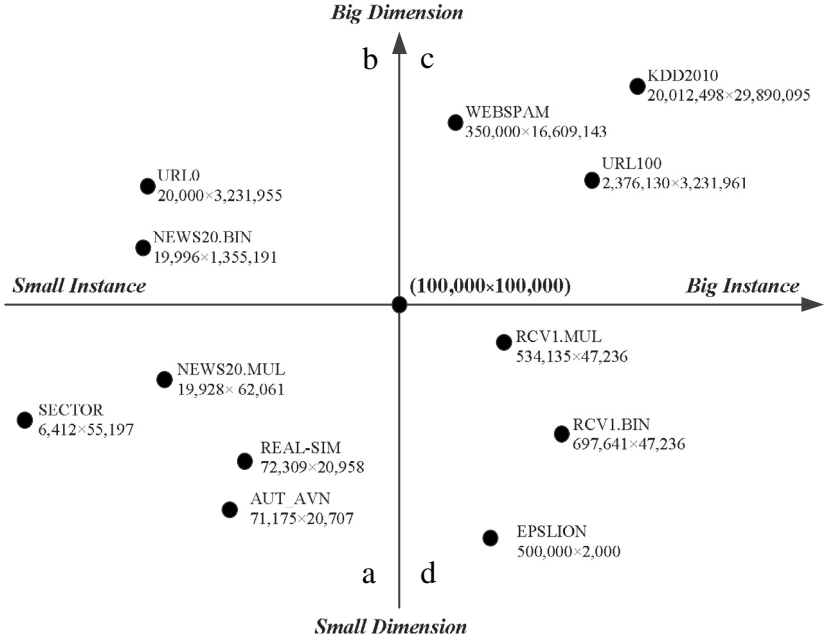


Fig. 3. Four groups of data: (a) *Small instance* and *Small dimension* (REAL-SIM, NEWS20.MUL, AUT_AVN, SECTOR); (b) *Small instance* and *Big dimension* (URL0, NEWS20.BIN); (c) *Big instance* and *Big dimension* (WEBSPAM, KDD2010, URL100); and (d) *Big instance* and *Small dimension* (RCV1.MUL, RCV1.BIN, EPSLION). $N \times d$: Instances × Dimensions. *Big dimension*: $d \geqslant 100{,}000$, *Big instance size*: $N \geqslant 100{,}000$.

5. Experimental study and performance evaluation

In this section, we evaluate the performance of the proposed method, termed FSVD-H-ELM, and assess it against several *state-of-the-art* methods on 12 datasets which belong to 4 cases as show in Fig. 3: Among them, all datasets except AUT_AVN[1] are obtained from the LIBSVM website.[2] URL0 and URL100 are the URL datasets from an anonymized 121-day subset of the ICML-09 URL data (Ma, Saul, Savage, & Voelker, 2009). The original URL dataset contains 121 independent subsets collected from 121 days. URL0 is the data from the first days using day-0 to predict day-1, while URL100 is the data from the first 100 days using them predict the rest days. These datasets have already been split into training and testing set as shown in Table 1.

All the experiments are conducted on a machine with the following configurations: MATLAB 2013a, Intel Xeon E5-2650, 2 GHz CPU (32 cores), 256G RAM. The comparison is conducted against other state-of-the-art large-scale machine learning algorithms including the Feature Generation Machine (FGM) (Tan, Tsang, & Wang, 2014), LibSVM (Chang & Lin, 2011), AMM (Wang et al., 2011), LLSVM (Zhang et al., 2012), BSGD (Wang et al., 2012) and others. FGM proposed recently in Tan et al. (2014) is used here for comparison, since it is one of the few *state-of-the-art* algorithms that has been designed to address feature selection and model learning of *Big dimensional* data and made available for download.[3] The main parameter of FGM is the number of selected features in each iteration. The larger selected features results in faster convergence while the smaller results in more accurate model. Here we set it to 100 for all problems as it can produce a good trading off between the efficiency and the effectiveness. For LibSVM, LLSVM and BSGD, the kernel is set to Gaussian kernel. The number of landmarks for LLSVM and the budgeted size of support vectors for BSGD-SVM is determined by cross-validation method. If the result is from other literature, it will be followed by the citation.

[1] http://vikas.sindhwani.org/svmlin.html.
[2] http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/.
[3] http://www.tanmingkui.com/publications.html.

Table 1
Details of datasets.

Datasets	# Instances	# Dimensions	# Classes
REAL-SIM	72,309	20,958	2
RCV1.BIN	697,641	47,236	2
RCV1.MUL	534,135	47,236	53
NEWS20.MUL	19,928	62,061	20
URL0	20,000	3,231,955	2
NEWS20.BIN	19,996	1,355,191	2
WEBSPAM	350,000	16,609,143	2
KDD2010	20,012,498	29,890,095	2
EPSILON	500,000	2,000	2
AUT_AVN	71,175	20,707	2
URL100	2,396,130	3,231,961	2
SECTOR	9,619	55,197	105

5.1. Performance comparison of FSVD-H-ELM, SVD-H-ELM and classical ELM

For the classical ELM, SVD-H-ELM and the proposed FSVD-H-ELM algorithm, the Sigmoid activation function $g(x) = \frac{1}{1+e^{-\lambda x}}$ is considered, where λ is set to '1' unless otherwise stated. The number of hidden nodes L is then chosen by cross-validation method with step size 100, and only the obtained optimized number of hidden nodes are reported. For FSVD-H-ELM, as the subset size $n = 200$ and $\rho = 0.5$ usually can achieve the promising generalized performance at the lowest training time, we would set $n = 200$ and $\rho = 0.5$ unless otherwise specified. The details about the optimized parameter for settings for every datasets are reported in Table 3.

In order to perform a comprehensive comparison, the experimental studies are carried out on three different training/testing partitions as shown in Table 2. For each partition, 10 trials (in each trial the data will be shuffled before partition to attain different training and testing set) are conducted, and then, the experimental results including the average performance and standard derivation (mean ± std.) of the proposed FSVD-H-ELM and *state-of-the-art* methods considered in the study are summarized in Tables 4, 6 and 8. To evaluate the performance difference of the considered algorithms, paired t-test (Rice, 2006) is introduced to obtain t value and significant level p. The smaller the p value the

Table 2
Three different training/testing partitions.

Datasets	Partition I		Partition II		Partition III	
	# Train set	# Test set	# Train set	# Test set	# Train set	# Test set
REAL-SIM	32,309	40,000	57,847	14,462	65,078	7,231
RCV1.BIN	677,399	20,242	558,110	139,531	627,880	69,761
RCV1.MUL	518,571	15,564	427,308	106,827	480,720	53,415
NEWS20.MUL	15,935	3,993	15,942	3,986	17,935	1,993
URL0	15,000	5,000	16,000	4,000	18,000	2,000
NEWS20.BIN	9,996	10,000	15,997	3,999	17,996	2,000
WEBSPAM	300,000	50,000	280,000	70,000	315,000	35,000
KDD2010	19,264,097	748,401	16,010,000	4,002,498	17,338,000	2,674,498
EPSILON	400,000	100,000	100,000	400,000	450,000	50,000
AUT_AVN	40,000	31,175	56,940	14,235	64,058	7,117
URL100	1,976,130	420,000	1,916,904	479,226	2,156,517	239,613
SECTOR	6,412	3,207	7,695	1,924	8,657	962

Table 3
Parameters of FSVD-H-ELM, SVD-H-ELM and classical ELM.

Datasets	FSVD-H-ELM						SVD-H-ELM			Classical ELM		
	P. I		P. II		P. III		P. I	P. II	P. III	P. I	P. II	P. III
	L	(n, ρ)	L	(n, ρ)	L	(n, ρ)	L	L	L	L	L	L
REAL-SIM	4000	(200,0.5)	4000	(200,0.5)	4000	(200,0.5)	4000	4000	4000	5500	5500	5500
RCV1.BIN	3000	(200,0.5)	3000	(200,0.5)	3000	(200,0.5)	3000	3000	3000	4000	4000	4000
RCV1.MUL	3000	(200,0.5)	3000	(200,0.5)	3000	(200,0.5)	3000	3000	3000	4500	4500	4500
NEWS20.MUL	1000	(200,0.5)	1000	(200,0.5)	1000	(200,0.5)	1000	1000	1000	2000	2000	2000
URL0	600	(200,0.5)	600	(200,0.5)	600	(200,0.5)	600	600	600	1000	1000	1000
NEWS20.BIN	2000	(200,0.5)	2000	(200,0.5)	2000	(200,0.5)	2000	2000	2000	2500	2500	2500
WEBSPAM	1500	(200,0.5)	1500	(200,0.5)	1500	(200,0.5)	1500	1500	1500	2500	2500	2500
KDD2010	150	(200,0.5)	150	(200,0.5)	150	(200,0.5)	150	150	150	350	350	350
EPSILON	600	(200,0.5)	600	(200,0.5)	600	(200,0.5)	600	600	600	1500	1500	1500
AUT_AVN	2000	(200,0.5)	2000	(200,0.5)	2000	(200,0.5)	2000	2000	2000	3000	3000	3000
URL100	2000	(200,0.5)	2000	(200,0.5)	2000	(200,0.5)	2000	2000	2000	3000	3000	3000
SECTOR	2000	(200,0.5)	2000	(200,0.5)	2000	(200,0.5)	2000	2000	2000	3000	3000	3000

L: the number of hidden nodes, n: the subset size, ρ: the selection ratio.

Table 4
Performance of FSVD-H-ELM, SVD-H-ELM and classical ELM on partition I.

Datasets	FSVD-H-ELM		SVD-H-ELM		Classical ELM	
	Training time (s)	Testing accuracy (%)	Training time (s)	Testing accuracy (%)	Training time (s)	Testing accuracy (%)
REAL-SIM	**26**	96.78 ± 0.046	153,413	**96.8** ± 0.044	44	93.30 ± 0.054
RCV1.BIN	1440	**97.10** ± 0.049		>48 h	**1102**	94.40 ± 0.053
RCV1.MUL	601	**88.54** ± 0.052		>48 h	**515**	85.92 ± 0.044
NEWS20.MUL	**45**	**84.94** ± 0.031	132,245	82.93 ± 0.024	79	81.07 ± 0.056
URL0	240	**97.18** ± 0.042	114,261	96.70 ± 0.041	**213**	95.43 ± 0.048
NEWS20.BIN	414	95.80 ± 0.050	104,169	**96.01** ± 0.045	**400**	87.85 ± 0.062
WEBSPAM	**8154**	**98.83** ± 0.034		*OOM*[a]	8321	93.46 ± 0.041
KDD2010	**1769**	**88.74** ± 0.047		*OOM*	1970	86.14 ± 0.055
EPSILON	452	**89.99** ± 0.054	34,678	89.31 ± 0.047	**365**	87.13 ± 0.063
AUT_AVN	42	**95.87** ± 0.048	14,822	95.48 ± 0.042	**41**	94.18 ± 0.052
URL100	610	**99.01** ± 0.043		>48 h	**589**	97.73 ± 0.051
SECTOR	19	**93.95** ± 0.054	108,334	93.88 ± 0.049	**11**	91.13 ± 0.063

[a] Out of memory.

Table 5
t value and significant level p of FSVD-H-ELM vs. SVD-H-ELM and classical ELM (Partition I).

	SVD-H-ELM (93.01%)	Classical ELM (90.64%)
FSVD-H-ELM (93.89%)	$t = 0.3714$, $p > 0.1$	$t = 1.6571$, $0.01 > p > 0.001$

more significant the difference. t value can be computed as follows:

$$t = \frac{\bar{\boldsymbol{a}}_i - \bar{\boldsymbol{a}}_j}{\sqrt{\frac{v_i^2}{n_i} + \frac{v_j^2}{n_j}}} \tag{37}$$

where $\boldsymbol{a}_i = a_{i,1}, a_{i,2}, \ldots, a_{i,12}$ denote the testing accuracies on the twelve datasets of ith algorithm, $\bar{\boldsymbol{a}}_i$ and $\bar{\boldsymbol{a}}_j$ denotes the mean value of $\boldsymbol{a}_i$ and $\boldsymbol{a}_j$, v_i^2 and v_j^2 represent the variance of $\boldsymbol{a}_i$ and $\boldsymbol{a}_j$, and n_i and n_j denotes the number of datasets (here $n_i = n_j = 12$). Then, by checking t-table (Rice, 2006), the significant level p is obtained by t value. The obtained t-test results are summarized in Tables 5, 7 and 9. With respect to the classical ELM, the training time incurred by FSVD-H-ELM and the classical ELM are observed to be competitive to one another on the twelve datasets. This may be attributed to the fact that, although the SVD operation of FSVD-H-ELM is originally thought to be more computationally intensive than the random assignment of hidden nodes in the classical ELM, it turns out that as the classical ELM often needs many more hidden nodes than is required by the FSVD-H-ELM, the overall computation time incurred by the classical ELM is equivalently high due to the need to perform the *Moore–Penrose generalized inverse* on a larger matrix (in the latter) when deriving the output weights. From Tables 5, 7 and 9, we can find that, on three partitions, the average testing accuracy, t-value and significant level p of FSVD-H-ELM and classical ELM are 93.89% vs. 90.64% ($t = 1.6571$, $0.01 > p >$

Table 6
Performance of FSVD-H-ELM, SVD-H-ELM and Classical ELM on Partition II.

Datasets	FSVD-H-ELM		SVD-H-ELM		Classical ELM	
	Training time (s)	Testing accuracy (%)	Training time (s)	Testing accuracy (%)	Training time (s)	Testing accuracy (%)
REAL-SIM	**46.4**	97.59 ± 0.045		>48 h	79.2	93.86 ± 0.053
RCV1.BIN	1187.4	**97.04** ± 0.046		>48 h	**937.2**	93.83 ± 0.054
RCV1.MUL	641	**88.65** ± 0.053		>48 h	**549.35**	86.12 ± 0.043
NEWS20.MUL	**72**	**86.33** ± 0.033	211,592	84.18 ± 0.022	126.40	82.37 ± 0.055
URL0	256.0	**97.42** ± 0.040	121,880	96.82 ± 0.040	**227**	95.87 ± 0.046
NEWS20.BIN	662.4	96.16 ± 0.051	166 670	96.21 ± 0.051	**640**	88.15 ± 0.060
WEBSPAM	**7583.2**	**98.81** ± 0.032		*OOM*[a]	7738.5	93.16 ± 0.042
KDD2010	**1470.2**	**88.72** ± 0.047		*OOM*	1637.3	86.11 ± 0.053
EPSILON	113	**89.12** ± 0.051	21,655	89.12 ± 0.046	**91.25**	86.87 ± 0.063
AUT_AVN	59.17	**95.98** ± 0.045	21 099	95.63 ± 0.043	**58.36**	94.32 ± 0.054
URL100	581.1	**89.97** ± 0.044		>48 h	**571.3**	97.75 ± 0.050
SECTOR	22.8	**94.12** ± 0.051	118,653	93.96 ± 0.051	**13.2**	91.17 ± 0.062

[a] Out of memory.

Table 7
t value and significant level p of FSVD-H-ELM vs. SVD-H-ELM and classical ELM (Partition II).

	SVD-H-ELM (92.65%)	Classical ELM (90.80%)
FSVD-H-ELM (93.33%)	$t = 0.3622$, $p > 0.1$	$t = 1.3487$, $0.01 > p > 0.001$

Table 9
t value and significant level p of FSVD-H-ELM vs. SVD-H-ELM and classical ELM (Partition III).

	SVD-H-ELM (92.4%)	Classical ELM (90.7%)
FSVD-H-ELM (94.02%)	$t = 0.4112$, $p > 0.1$	$t = 1.7223$, $0.01 > p > 0.001$

0.001), 93.33% vs. 90.80% ($t = 1.3487$, $0.01 > p > 0.001$), 94.02% vs. 90.7% ($t = 1.7223$ and $0.01 > p > 0.001$) respectively. This indicates that FSVD-H-ELM showcased significant improvements in the test performance with respect to classical ELM. FSVD-H-ELM outperformed classical ELM with 2%–8% improvements. For example, on the *Big dimensional* dataset `NEWS20.BIN` with Partition I, FSVD-H-ELM achieved a test performance of 95.80%, which is a significant improvement over the classical ELM accuracy of 87.85%. On the *Small dimensional* datasets `REAL-SIM` and `RCV1.BIN`, FSVD-H-ELM achieved test accuracy performances of 96.78% and 97.1%, respectively, which are also superior to that of the classical ELM at 93.3% and 94.4%, respectively. These indicate that the FSVD-H-ELM is able to appropriately capture the structural characteristic of the data much better than the classical ELM (which uses random hidden nodes).

With respect to the SVD-H-ELM, experimental results have only been successfully obtained on seven of the *small instance* datasets `REAL-SIM`, `NEWS20.MUL`, `URL0`, `NEWS20.BIN` and `AUT_AVN`. On the *Big instance* datasets, `RCV1.BIN`, `URL100` and `RCV1.MULT`, the simulations failed to complete despite 2 days of computations (48 h). On `WEBSPAM` and `KDD2010`, for instance, SVD-H-ELM went into the state of "out of memory (OOM)". From these results, it is clear that SVD-H-ELM is incapable of handling the *Big instance* dataset well as expected, thus the need for our introduction of the *divide-and-conquer approximation scheme* to arrive at the efficient FSVD-H-ELM. The results obtained indicate that FSVD-H-ELM is thousands of times more efficient than the SVD-H-ELM counterpart in most cases. For example, on `REAL-SIM`, SVD-H-ELM consumed a CPU wall clock time of 251,212 s to model the data, while FSVD-H-ELM only took 26 s to perform the same task. More importantly, the higher efficiency of FSVD-H-ELM is attained at no form of degradation in the test accuracy performance over SVD-H-ELM, in spite of the approximation. Interestingly, FSVD-H-ELM exhibits excellent test accuracy performance in most cases. As shown in Tables 5, 7 and 9, the average testing accuracy of FSVD-H-ELM vs. SVD-H-ELM on three partitions are 93.89% vs. 93.01% ($t = 0.3714$, $p > 0.1$), 93.33% vs. 92.65% ($t = 0.3622$, $p > 0.1$), and 94.02% vs. 92.4% ($t = 0.4112$ and $p > 0.1$) respectively. These showcased the improved test accuracies over the SVD-H-ELM. For example, on the `URL0` dataset with Partition I, FSVD-H-ELM achieved a test accuracy of 97.18%, which is slightly superior to the SVD-H-ELM of 96.70%. These results thus highlights the high efficacy of the FSVD-H-ELM in handling large-scale datasets. It is worth noting that the reasons for the reported high performances of the FSVD-H-ELM may be attributed to the ensemble like structure of FSVD-H-ELM, which integrates multiple SVD de-noised data subsets. Both ensemble and de-noising method have been proven to be effective for improving the resultant test accuracy performance in the literature (Kumar et al., 2009). FSVD-H-ELM thus inherits their benefits to generate the reported stable and excellent performances. Moreover, FSVD-H-ELM also eliminates the 'out of memory' problem that exists when dealing with *Big instance and Big dimensional* dataset such as `WEBSPAM` and `KDD2010` and shown

Table 8
Performance of FSVD-H-ELM, SVD-H-ELM and classical ELM on Partition III.

Datasets	FSVD-H-ELM		SVD-H-ELM		Classical ELM	
	Training time (s)	Testing accuracy (%)	Training time (s)	Testing accuracy (%)	Training time (s)	Testing accuracy (%)
REAL-SIM	**26**	97.98 ± 0.045		>48 h	90.4	94.21 ± 0.052
RCV1.BIN	1076.1	**97.08** ± 0.048		>48 h	**1022.3**	94.16 ± 0.055
RCV1.MUL	721.2	**88.54** ± 0.053		>48 h	**618**	85.92 ± 0.043
NEWS20.MUL	**81**	**84.94** ± 0.029	132,245	82.93 ± 0.023	142.2	81.07 ± 0.055
URL0	288	**97.18** ± 0.043	114,261	96.70 ± 0.040	**255.6**	95.43 ± 0.047
NEWS20.BIN	745.2	95.80 ± 0.050	187 500	**96.01** ± 0.046	**70**	87.85 ± 0.063
WEBSPAM	**8561.7**	**98.83** ± 0.032		*OOM*[a]	8737.1	93.46 ± 0.040
KDD2010	**1532.6**	**88.74** ± 0.049		*OOM*	1706.8	86.14 ± 0.057
EPSILON	508.5	**89.99** ± 0.052	39 013	89.31 ± 0.049	**410.625**	87.13 ± 0.064
AUT_AVN	67.26	**95.87** ± 0.049	23 737	95.48 ± 0.043	**65.6**	94.18 ± 0.053
URL100	663	**99.03** ± 0.042		>48 h	**642.7**	97.73 ± 0.052
SECTOR	25.6	**94.25** ± 0.055	119,312	93.98 ± 0.047	**14.85**	91.19 ± 0.062

[a] Out of memory.

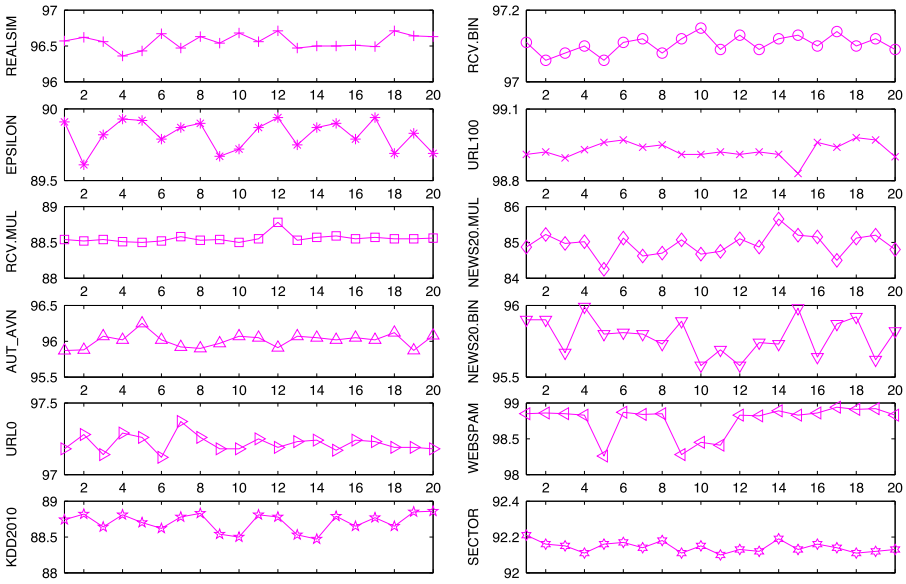


Fig. 4. FSVD-H-ELM testing accuracies of 20 trials on 12 datasets. In every trials the subsets are different. The testing accuracies are observed to remain stable across all 20 independent experimental trails.

to work with them efficiently and elegantly. It also showcased a test accuracy of 88.74% on the KDD2010 dataset that is not only superior to all the other *state-of-the-art* algorithms considered, it also completed the task within a short time span of 1769 s training time.

5.2. Performance comparison of FSVD-H-ELM to the other state-of-the-art machine learning methods

In this subsection, a comprehensive performance comparison of the FSVD-H-ELM is conducted against other *state-of-the-art* large-scale machine learning algorithms including the FGM (Tan et al., 2014), LLSVM, BSGD (Wang et al., 2012) and others. The performance results in terms of training time and testing accuracy are then summarized in Table 10. Here we reference and report the results from the relevant literatures if available. From the results, the following observations have been made:

1. With respect to the FGM, which is a recently proposed *state-of-the-art* algorithm for solving *Big dimensional* dataset, FSVD-H-ELM exhibited competitive generalization performances. Notably, although FGM is implemented in C++, FSVD-H-ELM is shown to incur shorter training time costs than FGM in many cases, despite the latter operating under a MATLAB environment. For example on NEWS20.BIN, FSVD-H-ELM took 414 s to achieve a test accuracy of 95.80%, while FGM took 454 s to achieve a similar accuracy of 95.08%. Furthermore, FGM is specially designed for two-class problems and cannot support multi-class problems directly, unless some extension is made. A possible solution is to first decompose the multi-class problem into multiple binary-class problems, train the classifiers to solve these problems, and then reconstruct the solution of the multi-class problem from the outputs attained by the classifiers. In contrast, FSVD-H-ELM can be convenient used for binary-class, multi-class and regression problems seamlessly.
2. With respect to the other *state-of-the-art* algorithms, FSVD-H-ELM has been observed to attain better generalization performances than TSVM (Sindhwani & Keerthi, 2006), LLSVM (Zhang et al., 2012), BSGD (Wang et al., 2012), DTSVM (Chang, Guo, Lin, & Lu, 2010) and AMM (Wang et al., 2011) on REAL-SIM, RCV1.MUL, NEWS20.MUL, NEWS20.BIN and KDD2010 at much lower training time. Although on the RCV1.BIN, URL0 and WEBSPAM datasets, FSVD-H-ELM did not perform significantly better than SVM (RBF), FGM and DTSVM, however, its test accuracies of 97.10% on RCV1.BIN, 97.18% on URL0 and 98.83% on WEBSPAM, are competitive to the optimal results of the *state-of-the-art* methods, which are 97.83%, 97.38% and 99.03%, respectively. More important, FSVD-H-ELM is able to attain the improved or competitive test accuracies at a much shorter training time than the *state-of-the-art* algorithms considered in most cases. In particular, on the *Big instance & Big dimensional* dataset KDD2010, which has more than 29 million features and 190 million training instance samples, FSVD-H-ELM is very competitive in terms of test accuracy. It showcased a testing accuracy of 88.74%, which is an improvement over the L1-SVM (88.4%), FGM (88.7%) and PROX-SLR (88.4%).

5.3. Stability of the divide and conquer fast approximation

In this subsection, the stability of the divide and conquer fast approximation scheme proposed is studied and visualized. For each dataset, 20 independent experimental trials have been conducted and the performance accuracies are summarized in Fig. 4. In all the trials, The parameter settings (including subset size and selection ratio) is set as in Table 3. The data subsets are constructed via random sampling from the original dataset and the SVD hidden nodes are derived subsequently. The results in Fig. 4 demonstrate excellent robustness of the proposed FSVD-H-ELM, where the testing accuracies are observed to remains stable across all 20 independent experimental trails.

5.4. The effects of data subset size n on SVD-H-ELM performance

Here we conduct further experimental study to evaluate the influence of data subset size n on FSVD-H-ELM performances, i.e., testing accuracy and computational time. Fig. 5 summarizes the testing accuracy and training time performances of the FSVD-H-ELM for different data subset size $n = [5, 10, 50, 200, 700, 1100, 1500, 3000]$ on the six datasets. From the figures, it is worth noting that the testing accuracy can be observed to remain relatively stable for different data subset size n, which highlights

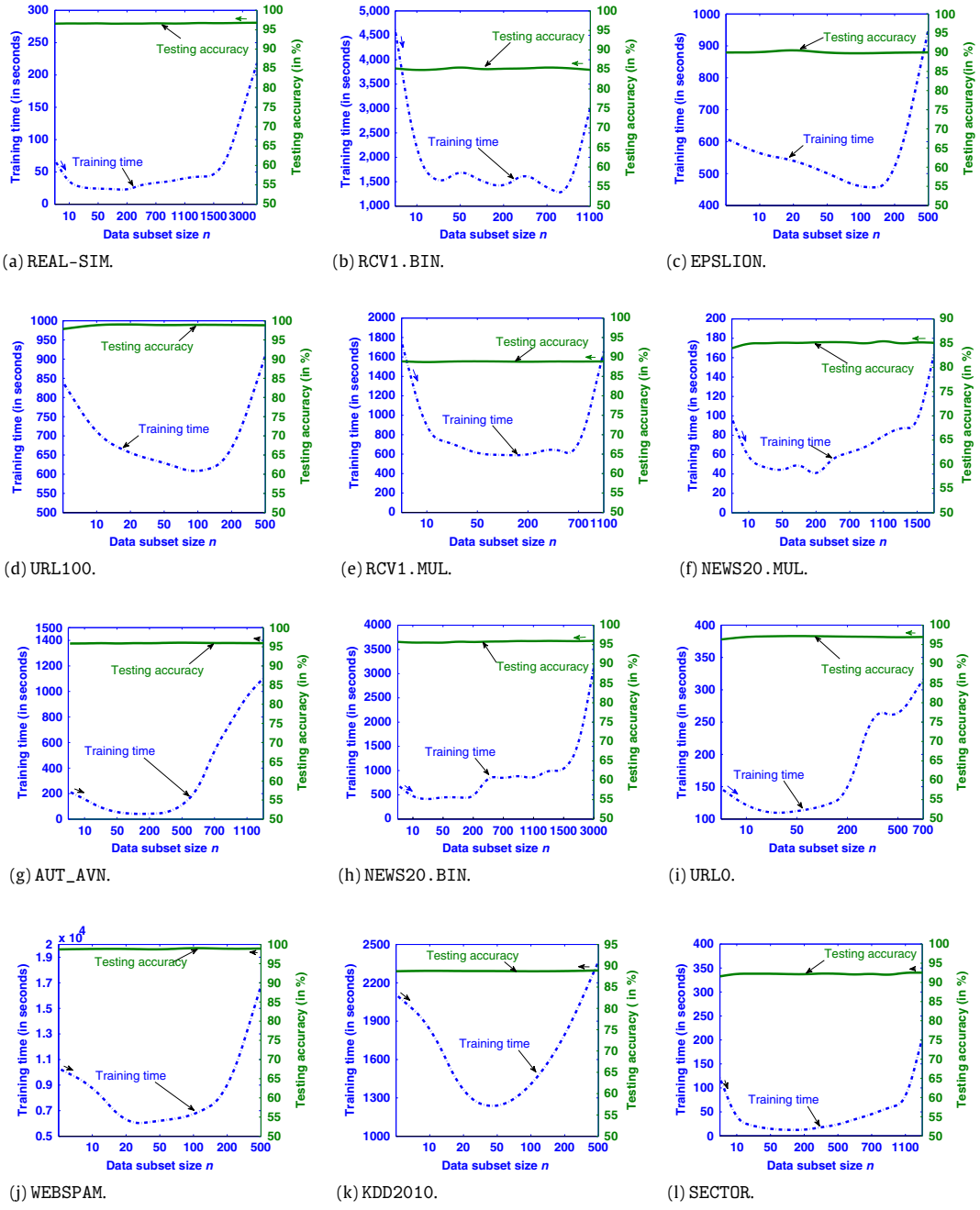
Table 10
Performance evaluation of FSVD-H-ELM and other state-of-the-art machine learning algorithms.

Datasets	Methods	Training time (s)	Testing accuracy (%)
Big instance and small dimension			
REAL-SIM	FSVD-H-ELM	**26**	**96.78**
	FGM	675	96.52
	TSVM (Sindhwani & Keerthi, 2006)	373	93.10
	DA (Sindhwani & Keerthi, 2006)	1,129	92.80
RCV1.BIN	FSVD-H-ELM	**1 440**	97.10
	FGM	10,923	97.42
	LLSVM (Deng et al., 2009)	1,800	95.77
	BSGD-SVM (Wang et al., 2012)	5,400	97.08
	LibSVM (RBF) (Choi et al., 2012)	72,720	**97.83**
EPSILON	FSVD-H-ELM	452	**89.89**
	CDN (Yuan et al., 2012)	3 000	89.82
	newGLMNET (Yuan et al., 2012)	500	89.82
	FGM	1,912	89.64
	LLSVM	3,870	88.71
	BSGD-SVM(5 pass)	6,896	89.58
URL100	FSVD-H-ELM	**640**	**98.97**
	AMM(Batch)	1 214	98.13
	AMM(Online)	1 465	98.10
	FGM	1,213	98.22
	LLSVM	2,512	98.17
	BSGD-SVM	1,701	98.32
Small instance and small dimension			
RCV1.MUL	FSVD-H-ELM	610	**88.54**
	K-Pegasos (Chen et al., 2011)	N.A.	84.50
	K-binaryLR (Chen et al., 2011)	N.A.	83.00
	Multi-class LR (Chen et al., 2011)	N.A.	88.50
	FGM	*Unsupported*[a]	
NEWS20.MUL	FSVD-H-ELM	**45**	**84.94**
	LASVM (Chang, Guo et al., 2010)	23,339	83.10
	DTSVM (Chang, Guo et al., 2010)	3,053	83.22
	CBD (Chang, Guo et al., 2010)	39,590	75.23
	Bagging (Chang, Guo et al., 2010)	35,176	76.55
	FGM	*Unsupported*[a]	
AUT_AVN	FSVD-H-ELM	**42**	**95.87**
	FGM	218	95.72
	AMM(Batch)	321	94.23
	AMM(Online)	346	94.10
	FGM	206	95.22
	LLSVM	531	94.97
	BSGD-SVM	318	95.32
Small instance and Big dimension			
NEWS20.BIN	FSVD-H-ELM	**414**	**95.80**
	FGM	454	95.08
	Forgetron (Orabona et al., 2009)	N.A.	90.00
	Projectron++ (Orabona et al., 2009)	N.A.	95.00
URL0	FSVD-H-ELM	240	97.18
	FGM	727	**97.38**
	AMM	**224**	97.13
Big instance and Big dimension			
WEBSPAM	FSVD-H-ELM	**8,154**	98.83
	CART (Chang, Guo et al., 2010)	29,332	98.44
	DTSVM (Chang, Guo et al., 2010)	63,015	**99.03**
KDD2010	FSVD-H-ELM	1,769	**88.74**
	L1-SVM (Tan et al., 2014)	1,500	88.40
	FGM (Tan et al., 2014)	1,870	88.70
	PROX-SLR (Tan et al., 2014)	**1,000**	88.40
SECTOR	FSVD-H-ELM	19	**93.95**
	LibSVM(OVA) (Rennie & Rifkin, 2001)	–	92.80
	NB(OVA) (Rennie & Rifkin, 2001)	–	64.30
	LibSVM(BCH63) (Rennie & Rifkin, 2001)	–	93.3
	NB(BCH63) (Rennie & Rifkin, 2001)	–	87.2
	FGM	*Unsupported*[a]	

[a] FGM is especially designed for two-class problems and cannot be applicable for multi-class problems directly. It needs some extending methods such as decomposing the multi-class problem to several two-class problems.

the robustness of the proposed FSVD-H-ELM. The training time, however, varies significantly for different subset size n. In particular, when n gets too large or too small, the computational time is noted to greatly increase. This can be comprehended with a complexity analysis of the FSVD-H-ELM.

Considering a data subset of size n, the original dataset of N data samples is first divided into $\lfloor \frac{N}{n} \rfloor$ data subsets $\{\mathbf{X}_r\}_{r=1}^{\lfloor \frac{N}{n} \rfloor}$. Assuming l hidden nodes are generated by each data subset of size n. A concatenation of the hidden nodes derived from each of the $s = \lceil \frac{L}{l} \rceil$ randomly chosen data subsets in $\{\mathbf{X}_r\}_{r=1}^{\lfloor \frac{N}{n} \rfloor}$ then



(a) REAL-SIM. (b) RCV1.BIN. (c) EPSLION.

(d) URL100. (e) RCV1.MUL. (f) NEWS20.MUL.

(g) AUT_AVN. (h) NEWS20.BIN. (i) URL0.

(j) WEBSPAM. (k) KDD2010. (l) SECTOR.

Fig. 5. Training time (in seconds) and Testing accuracy (in %) w.r.t. different data subset size n. In particular, the testing accuracy can be observed to remain relatively stable for different data subset sizes n. The training time, however, varies significantly with n.

forms a trained FSVD-H-ELM network of L hidden nodes. Note that the time complexity of SVD on a data subset of size n and dimension d is $\mathcal{O}\left(n^2 d\right)$, while other operations such as data subset retrieval takes a constant cost of Θ. Thus, FSVD-H-ELM has overall time complexity of $\mathcal{O}\left(\frac{L}{l}\cdot(n^2 d+\Theta)\right) \approx \mathcal{O}(Ld\cdot n + L\Theta\cdot\frac{1}{n})$. From the analysis, it is clear that when n gets too large, the first term of the complexity, i.e., the cost of SVD on s data subsets dominates. Conversely, when n gets too small, the second term, i.e., the data subset retrieval time dominates in comparison. As a result, the total computational cost of FSVD-H-ELM at $\mathcal{O}(Ld\cdot n + L\Theta\cdot\frac{1}{n})$ inflates when n is either too large or too small. In this manner, an appropriate data subset size that accords to the computational requirements of the user should be chosen. Based on our experimental study, a data subset size in the range of $n \in [50, 500]$ is noted to function well in general.

6. Conclusion

This paper has proposed the FSVD-H-ELM, which is designed to work with *Big dimensional* data. In contrast to existing approaches, we have embedded singular value decomposition hidden nodes into the classical ELM, while preserving the benefits of ELM where hidden nodes need not be tuned. In our study, SVD hidden nodes have been shown to be capable of capturing the structural properties of *Big dimensional* data, thus providing a more meaningful representation in the hidden layer of ELM. To address the high computational complexity of SVD, especially when dealing with large-scale problems, i.e., *Big instance* and *Big dimensional* data, a divide and conquer fast approximation scheme is made available in FSVD-H-ELM. The SVD hidden nodes which are derived from multiple random data subsets are used in place of the SVD hidden nodes derived from the entire dataset, thus

making it highly scalable to Big data. Comprehensive experiments have been conducted to validate the proposed FSVD-H-ELM on benchmark datasets. Further comparisons against several other *state-of-the-art* approaches verified and confirmed the superior generalization performance, robustness and efficiency of the proposed FSVD-H-ELM for solving *Big dimensional* data. Here we use SVD to obtain the hidden nodes, however, besides the use of SVD, it is worth noting that one may also choose other feature learning schemes including Non-negative Matrix Factorization (NMF), Kernel Principle Analysis (KPCA) and so on according to the problem of interest and requirement. This thus forms the immediate interest of our future works on Big Data analytics.

Acknowledgments

This work is supported by the ASTAR Thematic Strategic Research Programme (TSRP) Grant No. 1121720013, the Computational Intelligence Research Laboratory at NTU and National Science Foundation of China 61572399, 61532015, 61373116; Shaanxi New Star of Science & Technology 2013KJXX-29; New Star Team of Xian University of Posts & Telecommunications; Provincial Key Disciplines Construction Fund of General Institutions of Higher Education in Shaanxi.

References

Achlioptas, D., & Mcsherry, F. (2007). Fast computation of low rank matrix approximations. *Journal of the ACM*, *54*(2), 611–618.

Bai, Z., Huang, G.-B., Wang, D., Wang, H., & Westover, M. B. (2014). Sparse extreme learning machine for classification. *IEEE Transactions on Cybernetics*, *44*(10), 1858–1870.

Bingham, E., & Mannila, H. (2001). Random projection in dimensionality reduction: applications to image and text data. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining, KDD'01* (pp. 245–250).

Bordes, A., Bottou, L., & Gallinari, P. (2009). SGD-QN: Careful quasi-Newton stochastic gradient descent. *Journal of Machine Learning Research*, *10*, 1737–1754.

Bueno-Crespo, A., García-Laencina, P. J., & Sancho-Gómez, J.-L. (2013). Neural architecture design based on extreme learning machine. *Neural Networks*, *48*(0), 19–24.

Bullinaria, J. A., & AlYahya, K. (2014). Artificial bee colony training of neural networks: comparison with back-propagation. *Memetic Computing*, *6*(3), 171–182.

Butcher, J., Verstraeten, D., Schrauwen, B., Day, C., & Haycock, P. (2013). Reservoir computing and extreme learning machines for non-linear time-series data analysis. *Neural Networks*, *38*(0), 76–89.

Cambria, E., Huang, G.-B., Kasun, L. L. C., Zhou, H., Vong, C. M., Lin, J., … Liu, J. (2013). Extreme learning machines [trends controversies]. *IEEE Intelligent Systems*, *28*(6), 30–59. http://dx.doi.org/10.1109/MIS.2013.140.

Carl, E., & Gale, Y. (1936). The approximation of one matrix by another of lower rank. *Psychometrika*, *1*(3), 211–218.

Chacko, B. P., Krishnan, V. V., Raju, G., & Anto, P. B. (2012). Handwritten character recognition using wavelet energy and extreme learning machine. *International Journal of Machine Learning and Cybernetics*, *3*(2), 149–161.

Chang, F., Guo, C.-Y., Lin, X.-R., & Lu, C.-J. (2010). Tree decomposition for large-scale SVM problems. *Journal of Machine Learning Research*, *11*, 2935–2972.

Chang, Y.-W., Hsieh, C.-J., Chang, K.-W., Ringgaard, M., & Lin, C.-J. (2010). Training and testing low-degree polynomial data mappings via linear SVM. *Journal of Machine Learning Research*, *11*, 1471–1490.

Chang, C.-C., & Lin, C.-J. (2011). LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, *2*, 1–27.

Chaturvedi, I., Ong, Y.-S., & Arumugam, R. V. (2015). Deep transfer learning for classification of time-delayed Gaussian networks. *Signal Processing*, *110*, 250–262.

Chen, D., Chen, W., & Yang, Q. (2011). Characterizing inverse time dependency in multi-class learning. In *Proceedings of the 11th IEEE international conference on data mining, ICDM* (pp. 1020–1025).

Chen, H., Peng, J., Zhou, Y., Li, L., & Pan, Z. (2014). Extreme learning machine for ranking: Generalization analysis and applications. *Neural Networks*, *53*(0), 119–126.

Choi, K., Toh, K.-A., & Byun, H. (2012). Incremental face recognition for large-scale social network services. *Pattern Recognition*, *45*(8), 2868–2883.

Deng, W.-Y., Zheng, Q.-H., & Chen, L. (2009). Regularized extreme learning machine. In *Proceedings of IEEE symposium on computational intelligence and data mining* (pp. 389–395). IEEE.

Deng, W.-Y., Zheng, Q.-H., & Wang, Z.-M. (2014). Cross-person activity recognition using reduced kernel extreme learning machine. *Neural Networks*, *53*(0), 1–7.

Fan, J., Richard, S., & Wu, Y. (2009). Ultrahigh dimensional feature selection: beyond the linear model. *Journal of Machine Learning Research*, *10*, 2013–2038.

Feng, L., Ong, Y., Lim, M., & Tsang, I. (2014). Memetic search with inter-domain learning: A realization between cvrp and carp. *IEEE Transactions on Evolutionary Computation*, *19*(5), 644–658.

Fernandez-Delgado, M., Cernadas, E., Barro, S., Ribeiro, J., & Neves, J. (2014). Direct kernel perceptron (DKP): Ultra-fast kernel elm-based classification with non-iterative closed-form weight calculation. *Neural Networks*, *50*(0), 60–71.

Golub, Gene H., & Van Loan, Charles F. (2012). *Matrix computations. Vol. 3*. JHU Press.

Günter, S., Schraudolph, N. N., & Vishwanathan, S. V. N. (2007). Fast iterative kernel principal component analysis. *Journal of Machine Learning Research*, *8*, 1893–1918.

Halko, N., Martinsson, P.-G., & Tropp, J. A. (2011). Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Review*, *53*(2), 217–288.

Hanchuan, P., Fulmi, L., & Chris, D. (2005). Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *27*(8), 1226–1238.

Hsieh, C.-J., Chang, K.-W., Lin, C.-J., Keerthi, S. S., & Sundararajan, S. (2008). A dual coordinate descent method for large-scale linear SVM. In *Proceedings of the 25th international conference on Machine learning* (pp. 408–415). ACM.

Hsieh, C.-J., Si, S., & Dhillon, I. S. (2014). A divide-and-conquer solver for kernel support vector machines. In *Proceedings of the 31th international conference on machine learning* (pp. 566–574).

Huang, G.-B. (2014). An insight into extreme learning machines: Random neurons, random features and kernels. *Cognitive Computation*, *6*(3), 376–390.

Huang, G.-B. (2015). What are extreme learning machines? Filling the gap between Frank Rosenblatt's dream and John von Neumann's puzzle. *Cognitive Computation*, *7*(3), 263–278.

Huang, G.-B., Bai, Z., Kasun, L. L. C., & Vong, C. M. (2015). Local receptive fields based extreme learning machine. *IEEE Computational Intelligence Magazine*, *10*(2), 18–29.

Huang, G.-B., Chen, L., & Siew, C.-K. (2006). Universal approximation using incremental constructive feedforward networks with random hidden nodes. *IEEE Transactions on Neural Networks*, *17*(4), 879–892.

Huang, G.-B., Ding, X., & Zhou, H. (2010). Optimization method based extreme learning machine for classification. *Neurocomputing*, *74*(1), 155–163.

Huang, G., Huang, G.-B., Song, S., & You, K. (2015). Trends in extreme learning machines: A review. *Neural Networks*, *61*(0), 32–48.

Huang, G.-B., Zhou, H., Ding, X., & Zhang, R. (2012). Extreme learning machine for regression and multiclass classification. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, *42*(2), 513–529.

Huang, G.-B., Zhu, Q.-Y., & Siew, C.-K. (2006). Extreme learning machine: theory and applications. *Neurocomputing*, *70*(1), 489–501.

Jaeger, H. (2001). The "echo state" approach to analysing and training recurrent neural networks-with an erratum note, Bonn, Germany: German National Research Center for Information Technology GMD Technical Report 148 1–47.

Jaeger, H., & Haas, H. (2004). Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication. *Science*, *304*(5667), 78–80.

Jain, A. K., Duin, R. P. W., & Mao, J. (2000). Statistical pattern recognition: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *22*(1), 4–37.

Joachims, T. (2006). Training linear SVMs in linear time. In *Proceedings of the 12th ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 217–226). ACM.

Jordan, A. (2002). On discriminative vs. generative classifiers: A comparison of logistic regression and naive Bayes. *Advances in Neural Information Processing Systems*, *14*, 841.

Kan, E., Lim, M., Ong, Y., Tan, A., & Yeo, S. (2013). Extreme learning machine terrain-based navigation for unmanned aerial vehicles. *Neural Computing and Applications*, *22*(3–4), 469–477.

Kasun, L. L. C., Zhou, H., Huang, G.-B., & Vong, C. M. (2013). Representational learning with extreme learning machine for big data. *IEEE Intelligent Systems*, *28*(6), 1–4.

Krizhevsky, A., Sutskever, I., & Hinton, G.E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems, NIPS'2012* (pp. 1–9).

Kumar, S., Mohri, M., & Talwalkar, A. (2009). Ensemble Nystrom method. In *Advances in neural information processing systems 22, NIPS 2009* (pp. 1–9).

Kumar, S., Mohri, M., & Talwalkar, A. (2012). Sampling methods for the Nystrom method. *Journal of Machine Learning Research*, *13*(4), 981–1006.

Lin, S., Liu, X., Fang, J., & Xu, Z. (2015). Is extreme learning machine feasible? A theoretical assessment (part II). *IEEE Transactions on Neural Networks and Learning Systems*, *26*(1), 21–34.

Liu, X., Lin, S., Fang, J., & Xu, Z. (2015). Is extreme learning machine feasible? A theoretical assessment (part I). *IEEE Transactions on Neural Networks and Learning Systems*, *26*(1), 7–20.

Lowe, D. (1989). Adaptive radial basis function nonlinearities, and the problem of generalisation. In *First IEE international conference on artificial neural networks, 1989. (Conf. Publ. No. 313)* (pp. 171–175).

Luo, J., Vong, C.-M., & Wong, P.-K. (2014). Sparse Bayesian extreme learning machine for multi-classification. *IEEE Transactions on Neural Networks and Learning Systems*, *25*(4), 836–843.

Ma, J., Saul, L. K., Savage, S., & Voelker, G. M. (2009). Identifying suspicious URLs: an application of large-scale online learning. In *Proceedings of the 26th annual international conference on machine learning* (pp. 681–688). Montreal, Canada: ACM.

Maass, W., Natschläger, T., & Markram, H. (2002). Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural Computing*, *14*(11), 2531–2560.

Mao, Q., & Tsang, I. W. (2013). Efficient multi-template learning for structured prediction. *IEEE Transactions on Neural Networks and Learning Systems*, *24*(2), 248–261.

Mohammed, A., Minhas, R., Wu, Q. J., & Sid-Ahmed, M. (2011). Human face recognition based on multidimensional pca and extreme learning machine. *Pattern Recognition*, *44*(10), 2588–2597.

Mu, Y., Huay, G., Fan, W., & Chang, S.-F. (2014). Hash-SVM: Scalable kernel machines for large-scale visual classification. In *Conference on visualization and pattern recognition, CVPR* (pp. 1–8).

Orabona, F., Keshet, J., & Caputo, B. (2009). Bounded kernel-based online learning. *Journal of Machine Learning Research*, *10*, 2643–2666.

Paisitkriangkrai, S., Shen, C., & van den Hengel, A. (2014). A scalable stagewise approach to large-margin multiclass loss-based boosting. *IEEE Transactions on Neural Networks and Learning Systems*, *25*(5), 1002–1013.

Pao, Y. (1989). *Adaptive pattern recognition and neural networks*. Reading, MA, US: Addison-Wesley Publishing Co., Inc..

Pierre, C. (1994). Independent component analysis, a new concept? *Signal Processing*, *36*(3), 287–314.

Rahimi, A., & Recht, B. (2009). Weighted sums of random kitchen sinks: Replacing minimization with randomization in learning. In D. Koller, D. Schuurmans, Y. Bengio, & L. Bottou (Eds.), *Advances in neural information processing systems 21* (pp. 1313–1320). Curran Associates, Inc..

Rennie, J. D. M., & Rifkin, R. (2001). *Improving multiclass text classification with the support vector machine, Tech. rep*.

Rice, J. (2006). *Mathematical statistics and data analysis*. Cengage Learning.

Rong, H.-J., Ong, Y.-S., Tan, A.-H., & Zhu, Z. (2008). A fast pruned-extreme learning machine for classification problem. *Neurocomputing*, *72*(13), 359–366.

Rummelhart, D. (1986). Learning representations by back-propagation errors. *Nature*, *323*, 533–536.

Salama, M. A., Hassanien, A. E., & Revett, K. (2013). Employment of neural network and rough set in meta-learning. *Memetic Computing*, *5*(3), 165–177.

Saraswathi, S., Sundaram, S., Sundararajan, N., Zimmermann, M., & Nilsen-Hamilton, M. (2011). ICGA-PSO-ELM approach for accurate multiclass cancer classification resulting in reduced gene sets in which genes encoding secreted proteins are highly represented. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, *8*(2), 452–463.

Sarunas, R., & Vitalijus, P. (1980). On dimensionality, sample size, classification error, and complexity of classification algorithm in pattern recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *2*(3), 242–252.

Schaefer, G., Krawczyk, B., Celebi, M. E., & Iyatomi, H. (2014). An ensemble classification approach for melanoma diagnosis. *Memetic Computing*, *6*(4), 233–240.

Schmidt, W., Kraaijveld, M., & Duin, R. (1992). Feedforward neural networks with random weights. In *11th IAPR international conference on pattern recognition, 1992. Vol.II. conference B: pattern recognition methodology and systems, proceedings.* (pp. 1–4).

Seah, C.-W., Ong, Y.-S., & Tsang, I. (2013). Combating negative transfer from predictive distribution differences. *IEEE Transactions on Cybernetics*, *43*(4), 1153–1165.

Seah, C.-W., Tsang, I. W., & Ong, Y.-S. (2012). Transductive ordinal regression. *IEEE Transactions on Neural Networks and Learning Systems*, *23*(7), 1074–1086.

Seah, C.-W., Tsang, I. W., & Ong, Y.-S. (2013). Transfer ordinal label learning. *IEEE Transactions on Neural Networks and Learning Systems*, *24*(11), 1863–1876.

Shalev-Shwartz, S., Singer, Y., Srebro, N., & Cotter, A. (2011). Pegasos: Primal estimated sub-gradient solver for SVM. *Mathematical Programming*, *127*(1), 3–30.

Shalev-Shwartz, S., & Zhang, T. (2014). Accelerated proximal stochastic dual coordinate ascent for regularized loss minimization. In *Proceedings of the 31st international conference on machine learning* (pp. 64–72).

Sindhwani, V., & Keerthi, S. S. (2006). Large scale semi-supervised linear SVMs. In *Proceedings of the 29th annual international ACM SIGIR conference on research and development in information retrieval* (pp. 477–484). New York, NY, USA: ACM.

Tan, M., Tsang, I. W., & Wang, L. (2013). Minimax sparse logistic regression for very high-dimensional feature selection. *IEEE Transactions on Neural Networks and Learning Systems*, *24*(10), 1609–1622.

Tan, M., Tsang, I. W., & Wang, L. (2014). Towards ultrahigh dimensional feature selection for big data. *Journal of Machine Learning Research*, *15*, 1371–1429.

Tsang, I. W., Kwok, J. T., & Cheung, P.-M. (2005). Core vector machines: Fast svm training on very large data sets. *Journal of Machine Learning Research*, *6*, 363–392.

Verdaldi, A., & Zisserman, A. (2012). Efficient additive kernels via explicit feature maps. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *34*(3), 480–492.

Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., & Manzagol, P.-A. (2010). Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, *11*, 3371–3408.

Wang, Z., Djuric, N., Crammer, K., & Vucetic, S. (2011). Trading representability for scalability: Adaptive multi-hyperplane machine for nonlinear classification. In *Proccedings of the 17th ACM SIGKDD international conference on knowledge discovery and data mining, KDD'11* (pp. 24–32).

Wang, Z., Koby, C., & Slobodan, V. (2012). Breaking the curse of kernelization: Budgeted stochastic gradient descent for large-scale SVM training. *Journal of Machine Learning Research*, *13*(1), 3103–3131.

Widrow, B., Greenblatt, A., Kim, Y., & Park, D. (2013). The no-prop algorithm: A new learning algorithm for multilayer neural networks. *Neural Networks*, *37*, 182–188.

Xu, Y., Dong, Z. Y., Zhao, J. H., Zhang, P., & Wong, K. P. (2012). A reliable intelligent system for real-time dynamic security assessment of power systems. *IEEE Transactions on Power Systems*, *27*(3), 1253–1263.

Yu, H.-F., Hsieh, C.-J., Chang, K.-W., & Lin, C.-J. (2010). Large linear classification when data cannot fit in memory. In *Proceedings of the 16th ACM SIGKDD international conference on knowledge discovery and data mining, KDD'10* (pp. 833–842). New York, NY, USA: ACM.

Yuan, G.-X., Ho, C.-H., & Lin, C.-J. (2012). An improved GLMNET for l1-regularized logistic regression. *Journal of Machine Learning Research*, *13*(1), 1999–2030.

Zhai, Y., Ong, Y.-S., & Tsang, I. (2014). The emerging "big dimensionality". *IEEE Computational Intelligence Magazine*, *9*(3), 14–26.

Zhang, T. (2004). Solving large scale linear prediction problems using stochastic gradient descent algorithms. In *Proceedings of the twenty-first international conference on machine learning* (p. 116). ACM.

Zhang, K., Lan, L., Wang, Z., & Moerchen, F. (2012). Scaling up kernel SVM on limited resources: A low-rank linearization approach. In *Proceedings of the 15 th international conference on artificial intelligence and statistics, AISTATS, Vol. 22* (pp. 1425–1434).

Zhu, Z., Chen, W., Wang, G., Zhu, C., & Chen, Z. (2009). P-packSVM: Parallel primal gradient descent kernel SVM. In *Ninth IEEE international conference on data mining, 2009. ICDM'09* (pp. 677–686).