Soft Computing (2018) 22:3495–3506
https://doi.org/10.1007/s00500-018-3022-3

FOCUS

Discovering the impact of hidden layer parameters on non-iterative training of feed-forward neural networks

Zhiqi Huang[1] · Ran Wang[2] · Hong Zhu[3] · Jie Zhu[4]

Published online: 17 January 2018

Abstract
Considering restricted Boltzmann machine (RBM) as an unsupervised pre-training phase, this paper delivers a study on predetermined model parameters in extreme learning machine (ELM). Because of the non-iterative attribute in fine-tuning phase, the property of hidden layer output plays an important part in model performance. For ELM, we give a theoretical analysis on the hidden layer parameters related to matrix perturbation and continuity of generalized inverse. Then by empirically analyzing the proposed RBM–ELM algorithm, we find that the impact of hidden layer parameters on generalization ability varies among the experimental datasets. By exploring the training process and comparing the model parameters between random assignment and RBM, we identify the special pattern of hidden layer output discussed in theoretical part and empirically show that such pattern could harm the model performance.

Keywords Extreme learning machine · Restricted Boltzmann machine · Unsupervised pre-training

1 Introduction

Pre-training as a breakthrough to effective training strategies for deep architectures, in spite of different algorithms such as restricted Boltzmann machine (RBM) (Hinton et al. 2006) and autoencoder (Bengio et al. 2007), is all based on the following approach: an unsupervised training phase followed by a supervised fine-tuning phase. A comparison between Bernoulli RBM layers, stacked denoising autoencoders and standard feed-forward multilayer neural networks (Erhan et al. 2009, 2010) empirically shows that the unsupervised pre-training not only gives relatively good set of initials for tuning phase but also seems to act as a regularizer. And such properties consist with different network structures, size of datasets and order of observations, which demonstrate the importance of starting point in the non-convex optimization problem.

The existing research about unsupervised pre-training (Yu et al. 2010) is mainly based on the same fine-tuning method: back-propagation (BP) using gradient descent optimization algorithm and its improved versions. The tuning-based BP algorithm requires iterations for continuously changing the parameter according to loss function which often takes considerable amount of time and may have the cost-effective problem. In this paper, focusing on non-iterative tuning phase, we incorporate the idea of pre-training using RBM into extreme learning machines (ELMs) and evaluate the new RBM-ELM model. ELM, as a type of single hidden layer feed-forward neural networks (SLFNs) proposed in Huang et al. (2004, 2006), has been studied by many researches in recent decades (Wang et al. 2012, 2017b, 2018). Some researchers are focusing on various data types and training tasks (Ding et al. 2017a, b; Wang et al. 2015), and some stud-

Communicated by X. Wang, A.K. Sangaiah, M. Pelillo.

✉ Ran Wang
wangran@szu.edu.cn

Zhiqi Huang
huangzhiqi@szu.edu.cn

Hong Zhu
xszhuhong@163.com

Jie Zhu
arthurzhujie@163.com

[1] Guangdong Key Laboratory of Intelligent Information Processing, College of Computer Science and Software Engineering, Shenzhen University, Shenzhen, China

[2] College of Mathematics and Statistics, Shenzhen University, Shenzhen, China

[3] Faculty of Information Technology, Macau University of Science and Technology, Macao, China

[4] Department of Information Management, Central Institute for Correctional Police, Baoding, Hebei, China

ies have already shown that ELM is capable of handling big data (Mao et al. 2017; Zhai et al. 2017; Wang et al. 2017a; Li et al. 2017; Zhang et al. 2017). The training of the original ELM consists of two parts: first, the weights and bias between input and hidden layers are randomly assigned; second, the weights between hidden and output layers are obtained by solving a system of linear equations using generalized inverse (GI). Now, instead of random assigning, the RBM is used as an unsupervised pre-training phase for weights and bias between input and hidden layers; then, the weights between hidden and output layers are analytically solved by the generalized inverse.

Noting that such approach in SLFN is already mentioned in Pacheco et al. (2017), Chen et al. (2017) and extended to multiple-hidden layer feed-forward neural networks (MLFNs) in Wang et al. (2017c), Meng et al. (2017), we focus on, through extensive experimentation, how RBM affects the initial values of the network. And what causes the performance difference of RBM-ELM comparing with simple ELM across various datasets and sizes of hidden layer. Contrast from previous studies, the experimental results show that the generalization ability is not always improved by RBM pre-training. Following this line of phenomenon, we find that for some datasets, the trained RBM could lead to low variance of the hidden layer matrix. In such situation, the hidden layer matrix could be viewed as a constant matrix plus a perturbation, which would cause large variance of weights matrix between hidden and output layers due to discontinuity of the generalized inverse. Some theoretical analysis and explanations related to the matrix perturbation are given.

The rest of this paper is organized as follows. Section 2 lists a brief review on the concepts and algorithms of ELM and RBM. Section 3 discusses some theoretical proofs of generalized inverse. Details of RBM-ELM model and performance evaluation are given in Sect. 4. In Sect. 5, we conclude this paper.

2 Related work

2.1 Extreme learning machine

ELM means a three-layer feed-forward network with single hidden layer in which the weights and bias between input layer and hidden layer are randomly assigned and the weights between hidden layer and output layer are solved by a system of linear equations using generalized inverse. A simple structure of ELM for regression problem is shown in Fig. 1 with n nodes in input layer, m nodes in hidden layer and only one node in output layer, while for classification problem, the number of output nodes equals to the number of categories.

Given a set of samples $\mathbf{S} = \{(\mathbf{x}_i, \mathbf{t}_i)|\mathbf{x}_i \in \mathbf{R}^n, \mathbf{t}_i \in \mathbf{R}^d\}_{i=1}^N$, training process of ELM is to determine model parameters

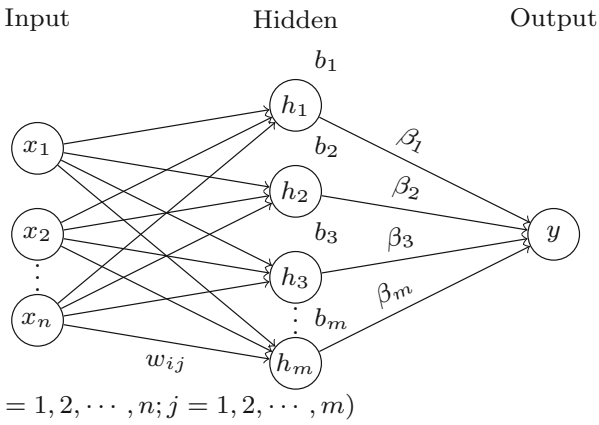

Fig. 1 A simple ELM structure

$\{w_{ij}, b_j, \beta_j\}$ $(i = 1, 2, \ldots, n;\ j = 1, 2, \ldots, m)$. Since the weights w_{ij} and bias b_j are randomly selected, the training process is only about determining the connections β_j between hidden layer and output layer. Let

$$\mathbf{G}_{N\times m} = \begin{bmatrix} \mathbf{w}_1\mathbf{x}_1 + b_1 & \cdots & \mathbf{w}_m\mathbf{x}_1 + b_m \\ \mathbf{w}_1\mathbf{x}_2 + b_1 & \cdots & \mathbf{w}_m\mathbf{x}_2 + b_m \\ \vdots & \ddots & \vdots \\ \mathbf{w}_1\mathbf{x}_N + b_1 & \cdots & \mathbf{w}_m\mathbf{x}_N + b_m \end{bmatrix} \tag{1}$$

be the middle matrix, where $\mathbf{w_j}$ is the jth column of the weight matrix $\mathbf{W}$ between input layer and hidden layer. Let $g(\cdot)$ be the sigmoid function and $\mathbf{H}$ be hidden layer matrix, then

$$\mathbf{H}_{N\times m} = [g(\mathbf{G})]_{N\times m} = [h_{ij}]_{N\times m} \tag{2}$$

Suppose the target matrix is $\mathbf{T} = [\mathbf{t}_1, \mathbf{t}_2, \ldots, \mathbf{t}_N]^T$, then the training of ELM is transferred to solve the system of linear equations $\mathbf{H}\boldsymbol{\beta} = \mathbf{T}$. In general, the solution $\mathbf{H}^-$ is not unique. It is suggested in Huang et al. (2006, 2012) to use the minimum-norm least-square solution. Instead of solving the system of linear equations, the optimization problem changes to:

$$\min_{||\boldsymbol{\beta}||}(\min_{\boldsymbol{\beta}\in\mathbf{R}^m} ||\mathbf{T} - \mathbf{H}\boldsymbol{\beta}||) \tag{3}$$

which means $\boldsymbol{\beta}_0$ is the solution of (3) if it has the smallest norm among all the least-square solutions. And the solution is Moore–Penrose generalized inverse $\mathbf{H}^\dagger$:

$$\mathbf{H}\boldsymbol{\beta} = \mathbf{T} \rightarrow \hat{\boldsymbol{\beta}} = \mathbf{H}^\dagger\mathbf{T} \tag{4}$$

Now the training process of an ELM can be viewed as three steps:

1. Increasing the data dimension from input data **S** to middle matrix **G**. In most cases, the number of hidden nodes m is greater than number of input attributes n;
2. Transferring middle matrix **G** to hidden layer matrix **H** with rank increased by sigmoid activation function;
3. Solving a system of linear equations with full rank of coefficient matrix.

In the following, we give some propositions and remarks regarding the ELM training phase.

Proposition 1 *Assume that* $\mathbf{v} = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_N\}$, $\mathbf{v}_i = \{v_{i1}, v_{i2}, \ldots, v_{in}\}$, $i = 1, 2, \ldots, N$ *denotes a set of n-dimensional vectors, such that* $1 \leq rank(\mathbf{v}) \leq n$. *Then with probability 1, the sigmoid transformation will transfer* $\mathbf{v}$ *into a set of vectors of full rank:*

$$rank(\mathbf{H}) = n \quad w.p.1 \tag{5}$$

where $\mathbf{H} = \{\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_N\}$, $\mathbf{h}_i = \{h_{i1}, h_{i2}, \ldots, h_{in}\}$, $h_{ij} = sigmoid(v_{ij}) = 1/(1 + e^{v_{ij}})$, $i = 1, 2, \ldots, N$, $j = 1, 2, \ldots, n$.

Proposition 2 *The generalized inverse* $\mathbf{H}^{\dagger}$ *is continuous if* $\mathbf{H}$ *is a full rank matrix.*

The proof for Propositions 1 and 2 can be found in Fu et al. (2014). According to the ELM training process and the two important propositions, we have the following remarks

Remark 1 In step 2, the middle matrix **G** is coming from input data **S** via a linear transformation and is generally waning rank. Proposition 1 guarantees that the sigmoid transformation will transfer a waning rank matrix **G** to a full rank matrix **H**.

Remark 2 In step 3, to reach the minimum-norm least-square estimator of $\boldsymbol{\beta}$, the generalized inverse of a full rank matrix is calculated. Proposition 2 guarantees the stability of the solution.

Remark 3 Since the input matrix of ELM is transferred to a middle matrix by a group of random parameters, the relationship between the perturbation of weights and sensitivity of solution measures the stability of the model. According to the experimental result in Fu et al. (2014, Section 5), the full rank matrix **H** is insensitive to the perturbation and can get a more stable solution for $\mathbf{H}\boldsymbol{\beta} = \mathbf{T}$.

2.2 Restricted Boltzmann machine

RBM is a generative stochastic model which can be used to capture the probability distribution over a set of inputs (Hinton et al. 2012). Figure 2 shows the structure of a RBM. The topology of RBM is a two-layer bipartite graph: the

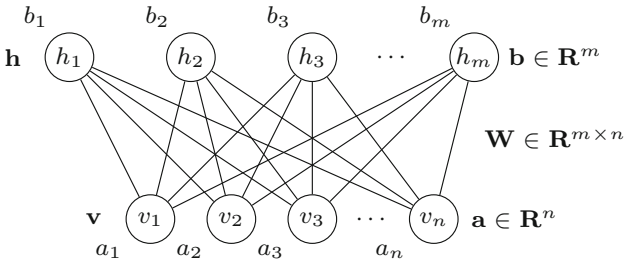

Fig. 2 Structure of a RBM

underlying visible layer, denoted as $\mathbf{v} = [v_1, v_2, \ldots, v_n]$, is used to receive the input data and the upper hidden layer, denoted as $\mathbf{h} = [h_1, h_2, \ldots, h_m]$, is used to generate a new vector based on visible layer and training algorithm.

Suppose a training set $S = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_N\}$ that contains N observations, where $\mathbf{v}^{\tau} = [v_{\tau 1}, v_{\tau 2}, \ldots, v_{\tau n}]$, $\tau = 1, \ldots, N$ is the τth sample observation. Then, the energy (Smolensky 1986) of a RBM configuration is defined as following:

$$E_{\theta}(\mathbf{v}, \mathbf{h}) = -\sum_{j=1}^{n} a_j v_j - \sum_{i=1}^{m} b_i h_i - \sum_{j=1}^{n}\sum_{i=1}^{m} w_{ij} v_j h_i \tag{6}$$

where $\mathbf{a} = [a_1, a_2, \ldots, a_n] \in \mathbf{R}^n$ is the visible layer bias, $\mathbf{b} = [b_1, b_2, \ldots, b_m] \in \mathbf{R}^m$ is the hidden layer bias, $\mathbf{W} = [w_{ij}] \in \mathbf{R}^{m \times n}$ is the weight matrix and $\theta = \mathbf{W}, \mathbf{a}, \mathbf{b}$ represents the set of model parameters. According to (6), the probability of system being in current status can be obtained by

$$p_{\theta}(\mathbf{v}, \mathbf{h}) = \frac{1}{Z_{\theta}} e^{-E_{\theta}(\mathbf{v}, \mathbf{h})} \tag{7}$$

where $Z_{\theta} = \sum_{\mathbf{v},\mathbf{h}} E_{\theta}(\mathbf{v}, \mathbf{h})$ is the normalization factor. From joint distribution (7), we have $\mathbf{v}$'s marginal probability distribution

$$p_{\theta}(\mathbf{v}) = \frac{1}{Z_{\theta}} \sum_{\mathbf{h}} e^{-E_{\theta}(\mathbf{v}, \mathbf{h})}$$

The object of training RBM is to minimize the Kullback–Leibler divergence from the model distribution p_{θ} to the true distribution of the data p_t, i.e., $D_{KL}(p_t || p_{\theta})$. And this is equivalent to maximizing the log likelihood function

$$\mathcal{L}(\theta; \mathbf{v}) = \sum_{\mathbf{v}} \ln \frac{1}{Z_{\theta}} \sum_{\mathbf{h}} e^{-E(\theta; \mathbf{v}, \mathbf{h})}$$

Assume the data in visible and hidden layers are all subjected to Bernoulli distribution, then value of each node is in $\{0, 1\}$. For numerical attributes, one can refer to the Gaussian–Bernoulli RBM in Hinton (2010), Salakhutdinov

et al. (2007). Now apply gradient descent method with respect to θ; we have the following:

$$\begin{aligned}
\frac{\partial \mathcal{L}(\theta;\mathbf{v})}{\partial w_{ij}} &= \sum_{\tau=1}^{N}\left[p(h_i=1|\mathbf{v}^{\tau})v_j^{\tau} - \sum_{\mathbf{v}} p(\mathbf{v})p(h_i=1|\mathbf{v})v_j\right]\\
\frac{\partial \mathcal{L}(\theta;\mathbf{v})}{\partial a_j} &= \sum_{\tau=1}^{N}\left[v_j^{\tau} - \sum_{\mathbf{v}} p(\mathbf{v})v_j\right]\\
\frac{\partial \mathcal{L}(\theta;\mathbf{v})}{\partial b_i} &= \sum_{\tau=1}^{N}\left[p(h_i=1|\mathbf{v}^{\tau}) - \sum_{\mathbf{v}} p(\mathbf{v})p(h_i=1|\mathbf{v})\right]
\end{aligned} \tag{8}$$

where the conditional probability is given by

$$\begin{cases}
p(h_i=1|\mathbf{v}) = \text{sigmoid}\left(b_i + \sum_{j=1}^{n} w_{ij}v_j\right)\\
p(v_j=1|\mathbf{h}) = \text{sigmoid}\left(a_j + \sum_{i=1}^{m} w_{ij}h_i\right)
\end{cases}$$

Derived from Z_θ, on the right-hand side of (8), $\sum p(\mathbf{v})$ shows up in all three equations, and the computational complexity is $O(2^{m+n})$. Due to the number of visible and hidden nodes, it is hard to update the parameters based on these gradient formulas. An efficient approximation method named contrastive divergence (CD) was introduced by Hinton (2006). The main idea of k step CD is that instead of minimizing $D_{KL}(p_t||p_\theta)$, the method minimizes the difference between p_t and p_θ^k, i.e., $D_{KL}(p_t||p_\theta^k)$, where p_θ^k is the distribution over the k step reconstructions of the data vectors generated by Gibbs sampling. In practice, we usually choose $k=1$; then, the CD-1 for updating gradient descent is listed below

$$\begin{aligned}
\frac{\partial \mathcal{L}(\theta;\mathbf{v})}{\partial w_{ij}} &\approx \sum_{\tau=1}^{N}\Big[p(h_i=1|\mathbf{v}^{(\tau,0)})v_j^{(\tau,0)}\\
&\quad - p(h_i=1|\mathbf{v}^{(\tau,1)})v_j^{(\tau,1)}\Big]\\
\frac{\partial \mathcal{L}(\theta;\mathbf{v})}{\partial a_j} &\approx \sum_{\tau=1}^{N}\left[v_j^{(\tau,0)} - v_j^{(\tau,1)}\right]\\
\frac{\partial \mathcal{L}(\theta;\mathbf{v})}{\partial b_i} &\approx \sum_{\tau=1}^{N}\left[p(h_i=1|\mathbf{v}^{(\tau,0)}) - p(h_i=1|\mathbf{v}^{(\tau,1)})\right]
\end{aligned} \tag{9}$$

where $\mathbf{v}^{(\tau,0)}$ is the starting point of sampling from training dataset and $\mathbf{v}^{(\tau,1)}$ is the sampled point using CD-1 algorithm. We can use (9) as the gradient formulas.

3 Perturbation of generalized inverse

In this section, we will give some new theoretical findings for the ELM training phase, based on the preliminaries introduced in Sect. 2.

From Proposition 2, we already know that the generalized inverse is continuous if the matrix is full rank. Now we focus on a special case: the matrix $\mathbf{H}$ is still full rank, but the values inside $\mathbf{H}$ are all near a constant. We have two aims in this section. One is to prove the discontinuity of generalized inverse if matrix $\mathbf{H}$ is not full rank. Second is to discuss the pattern and trends of hidden layer output when generalized inverse is discontinuous. Because generalized inverse is closely related to the singular value decomposition (SVD), starting from CourantFischer min–max theorem, we will prove the perturbation rule of singular value.

Proposition 3 (CourantFischer) *Suppose* $\mathbf{A}$ *is a* $n \times n$ *real symmetric matrix, let* $\{\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_r\}$, $r = 0, 1, \ldots, n-1$, *denote the set of* n*-dimensional vectors. Also, enumerate the* n *eigenvalues of* $\mathbf{A}$, $\lambda_1, \lambda_2, \ldots, \lambda_n$ *in increasing order, i.e.,* $\lambda_1 \le \cdots \le \lambda_n$. *Then, we have*

$$\min_{\{\mathbf{p}_i\}} \max_{\substack{||\mathbf{x}||=1\\ \mathbf{x}^T\mathbf{p_i}=0}} \mathbf{x}^T\mathbf{A}\mathbf{x} = \lambda_{r+1} \tag{10}$$

$$\max_{\{\mathbf{p}_i\}} \min_{\substack{||\mathbf{x}||=1\\ \mathbf{x}^T\mathbf{p_i}=0}} \mathbf{x}^T\mathbf{A}\mathbf{x} = \lambda_{n-r} \tag{11}$$

The proof of Proposition 3 can be found in Parlett (1998). Now suppose we have a $m \times n$ matrix $\mathbf{A}$ with $rank(\mathbf{A}) = k$, the norm of $\mathbf{A}$ and its generalized inverse $\mathbf{A}^\dagger$ can be represented by its singular values.

Proposition 4 *Suppose the singular values of* $\mathbf{A}^{m\times n}$ *are* $\lambda_1 \ge \lambda_2 \ge \cdots \ge \lambda_k > 0$, *then*

$$||\mathbf{A}|| = \lambda_1 \quad \text{and} \quad ||\mathbf{A}^\dagger|| = \lambda_k^{-1} \tag{12}$$

Proof The definition of norm

$$||\mathbf{A}|| = \max_{||\mathbf{x}||=1} ||\mathbf{A}\mathbf{x}||, \quad \mathbf{x} = \mathbf{R}^n$$

According to the definition of Euclidean norm

$$||\mathbf{A}\mathbf{x}||^2 = \mathbf{x}^T\mathbf{A}^T\mathbf{A}\mathbf{x}$$

The eigenvalues of $\mathbf{A}^T\mathbf{A}$ are $\lambda_1^2 \ge \lambda_2^2 \ge \cdots \ge \lambda_k^2$ and eigenvectors $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k$, so

$$\max_{||\mathbf{x}||=1} ||\mathbf{Ax}||^2 = \max_{||\mathbf{x}||=1} \left(\mathbf{x}^T\mathbf{A}^T\mathbf{Ax}\right)$$
$$= \max_{||\mathbf{x}||=1} \left(\mathbf{x}^T \sum_1^k \lambda_i \mathbf{v}_i \mathbf{v}_i^T\right)\mathbf{x}$$
$$= \max_{||\mathbf{x}||=1} \sum_1^k \lambda_i^2 \left(\mathbf{x}^T\mathbf{v}_i\right)^2$$

with $\sum_1^k (\mathbf{x}^T\mathbf{v}_i)^2 \le 1$, then $\max_{||\mathbf{x}||=1} ||\mathbf{Ax}||^2 \le \lambda_1^2$. If let $\mathbf{x} = \mathbf{v}_1$, then

$$\mathbf{x}^T\mathbf{A}^T\mathbf{Ax} = \lambda_1^2 \leftrightarrow \max_{||\mathbf{x}||=1} ||\mathbf{Ax}||^2 = \lambda_1^2 \leftrightarrow ||\mathbf{A}|| = \lambda_1$$

Now consider the $||\mathbf{A}^\dagger||$. Assume $\mathbf{A}$ has singular value decomposition (SVD) $\mathbf{A} = \mathbf{U\Sigma V}^T$, then $\mathbf{A}^\dagger = \mathbf{V\Sigma}^\dagger\mathbf{U}^T$, where

$$\mathbf{\Sigma} = \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_k \end{bmatrix} \quad \text{and} \quad \mathbf{\Sigma}^{-1} = \begin{bmatrix} \lambda_1^{-1} & & \\ & \ddots & \\ & & \lambda_k^{-1} \end{bmatrix}$$

$$||\mathbf{A}^\dagger||^2 = \max_{||\mathbf{x}||=1} ||\mathbf{A}^\dagger\mathbf{x}||^2$$
$$= \max_{||\mathbf{x}||=1} \left\{\left(\mathbf{V\Sigma}^{-1}\mathbf{U}^T\mathbf{x}\right)^T \left(\mathbf{V\Sigma}^{-1}\mathbf{U}^T\mathbf{x}\right)\right\}$$
$$= \max_{||\mathbf{y}||=1} \mathbf{y}^T\mathbf{\Sigma}^{-2}\mathbf{y}$$

Same as the norm of $\mathbf{A}$, the norm of $\mathbf{A}^\dagger$ is the square root of the largest eigenvalue of $\mathbf{\Sigma}^{-2}$ which is λ_k^{-1}.

Suppose a small perturbation $\boldsymbol{\delta}\mathbf{A}$ and $\mathbf{B} = \mathbf{A} + \boldsymbol{\delta}\boldsymbol{\delta}\mathbf{A}$. Regarding the singular values of $\mathbf{A}$ and $\mathbf{B}$, we have the following proposition. □

Proposition 5 *Suppose* $rank(\mathbf{A}) = rank(\mathbf{B}) = k$ *and the singular values of* $\mathbf{A}$ *are* $\lambda_1 \ge \lambda_2 \ge \cdots \ge \lambda_k$, *because* $\mathbf{B}$ *has the same rank with* $\mathbf{A}$, $\mathbf{B}$ *has singular values* $\sigma_1 \ge \sigma_2 \ge \cdots \ge \sigma_k$. *Then,*

$$\sigma_i \le \lambda_i + ||\boldsymbol{\delta}\mathbf{A}|| \tag{13}$$

Proof According to the singular value decomposition (SVD), $\mathbf{A}^T\mathbf{A}$ has the eigenvalues $\lambda_1^2, \lambda_2^2, \ldots, \lambda_k^2$ and eigenvector $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k$; then, apply the min–max rule in Proposition 3; let $\mathbf{v}_i = \mathbf{p}_i$, we have

$$\sigma_{r+1}^2 \le \max_{\substack{||\mathbf{x}||=1 \\ \mathbf{x}^T\mathbf{p_i}=0}} \mathbf{x}^T\mathbf{B}^T\mathbf{Bx}$$
$$= \max_{\substack{||\mathbf{x}||=1 \\ \mathbf{x}^T\mathbf{p_i}=0}} \mathbf{x}^T\left(\mathbf{A}+\boldsymbol{\delta}\mathbf{A}\right)^T\left(\mathbf{A}+\boldsymbol{\delta}\mathbf{A}\right)\mathbf{x}$$
$$\le \max_{\substack{||\mathbf{x}||=1 \\ \mathbf{x}^T\mathbf{p_i}=0}} \left\{\left(\mathbf{x}^T\mathbf{A}^T\mathbf{Ax}\right)^{\frac{1}{2}} + \left(\mathbf{x}^T\left(\boldsymbol{\delta}\mathbf{A}\right)^T\left(\boldsymbol{\delta}\mathbf{A}\right)\mathbf{x}\right)^{\frac{1}{2}}\right\}^2$$
$$\le \Bigg\{ \max_{\substack{||\mathbf{x}||=1 \\ \mathbf{x}^T\mathbf{p_i}=0}} \left(\mathbf{x}^T\mathbf{A}^T\mathbf{Ax}\right)^{\frac{1}{2}}$$
$$+ \max_{\substack{||\mathbf{x}||=1 \\ \mathbf{x}^T\mathbf{p_i}=0}} \left(\mathbf{x}^T\left(\boldsymbol{\delta}\mathbf{A}\right)^T\left(\boldsymbol{\delta}\mathbf{A}\right)\mathbf{x}\right)^{\frac{1}{2}} \Bigg\}^2$$
$$\le \left(\lambda_{r+1} + ||\delta\mathbf{A}||\right)^2, \quad r = 1, 2 \ldots, k-1.$$

Thus

$$\sigma_{r+1} \le \lambda_{r+1} + ||\boldsymbol{\delta}\mathbf{A}|| \leftrightarrow \sigma_r \le \lambda_r + ||\boldsymbol{\delta}\mathbf{A}||$$

□

Proposition 6 *If the* $m \times n$ $(m < n)$ *matrix* $\mathbf{A}$ *is waning rank,* $rank(\mathbf{A}) = k < n$, *the small perturbation* $\boldsymbol{\delta}\mathbf{A}$ *increases the rank of* $\mathbf{B} = \mathbf{A} + \boldsymbol{\delta}\mathbf{A}$.

$$rank(\mathbf{A} + \boldsymbol{\delta}\mathbf{A}) > rank(\mathbf{A}) = k \tag{14}$$

Then, we have the inequation:

$$||(\mathbf{A} + \boldsymbol{\delta}\mathbf{A})^\dagger|| \ge \frac{1}{||\boldsymbol{\delta}\mathbf{A}||} \tag{15}$$

Proof Assume $rank(\mathbf{A}+\boldsymbol{\delta}\mathbf{A}) = r > k$, then the rth singular value of matrix A is $\lambda_r = 0$. According to Proposition 5, the rth singular value of $\mathbf{A} + \boldsymbol{\delta}\mathbf{A}$, σ_r has

$$\sigma_r \le ||\boldsymbol{\delta}\mathbf{A}||$$

Meanwhile, apply Proposition 4, the norm of $(\mathbf{A} + \boldsymbol{\delta}\mathbf{A})^\dagger$ has

$$||(\mathbf{A} + \boldsymbol{\delta}\mathbf{A})^\dagger|| \le \frac{1}{\sigma_r}$$

Therefore

$$||(\mathbf{A} + \boldsymbol{\delta}\mathbf{A})^\dagger|| \ge \frac{1}{||\boldsymbol{\delta}\mathbf{A}||}$$

□

Example 1 Let $\mathbf{H} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$, then $rank(\mathbf{H}) = 1$, which is not full rank. It is easy to calculate that $\mathbf{H}^\dagger = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$.

Suppose that $\boldsymbol{\delta}\mathbf{H} = \begin{bmatrix} 0 & 0 \\ 0 & \epsilon \\ 0 & 0 \end{bmatrix}, \epsilon \neq 0$, then $\mathbf{H} + \boldsymbol{\delta}\mathbf{H} = \begin{bmatrix} 1 & 0 \\ 0 & \epsilon \\ 0 & 0 \end{bmatrix}$. Noting that the rank is increased from 1 to 2 and $\mathbf{H} + \delta\mathbf{H}$ is full rank, we get $(\mathbf{H} + \boldsymbol{\delta}\mathbf{H})^{\dagger} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \epsilon^{-1} & 0 \end{bmatrix}$. It is easy to see that limit of $(\mathbf{H} + \delta\mathbf{H})^{\dagger}$ does not exists when $\epsilon \to 0$. And the variance will increase when ϵ decreases.

Remark 4 (15) and Example 1 demonstrate that if A is a waning rank matrix and a perturbation of $\mathbf{A}$ increases the rank of $\mathbf{A} + \boldsymbol{\delta}\mathbf{A}$, the smaller of $||\boldsymbol{\delta}\mathbf{A}||$, the larger $||(\mathbf{A} + \boldsymbol{\delta}\mathbf{A})^{\dagger}||$ will be. When $||\boldsymbol{\delta}\mathbf{A}|| \to 0$, $||(\mathbf{A} + \boldsymbol{\delta}\mathbf{A})^{\dagger}|| \to \infty$, the generalized inverse does not exist. We conclude that the generalized inverse $\mathbf{H}^{\dagger}$ is discontinuous if $\mathbf{H}$ is waning rank. This conclusion is related to the continuity of singular value. For diagonal matrix $\boldsymbol{\Sigma}$, we get the generalized inverse by taking the reciprocal of each nonzero element on the diagonal, leaving the zeros in place and then transposing the matrix.

Remark 5 In the next section, we will show that for some datasets, the RBM pre-trained hidden layer outputs are close to each other. Although the hidden layer matrix is still full rank, but because of the low variance, the matrix could be viewed a perturbation around a constant. Thus, when solving the connections between hidden and output layers, the generalized inverse is decided by the scope of perturbation, not the value itself. Named as "pseudo-full rank," this degenerating of hidden layer output has a negative impact on model generalization ability. The situation might not only happen to unsupervised pre-training but also for random initiating. That means, monitoring the variance of hidden layer matrix is necessary when work with ELM.

4 Algorithm and experiments

In this section, we will conduct some empirical studies to validate the theoretical findings in Sect. 3. More specifically, we determine the hidden layer of an ELM by unsupervised pre-training strategy. Instead of randomly selecting weights and bias, now we train a RBM based on input data S, number of visible nodes n and number of hidden nodes m. The training steps of RBM-ELM are given in Algorithm 1. Then, some benchmark datasets with different sizes and number/type of attributes are chosen from UCI Machine Learning Repository (Lichman 2013) and handwritten databases (LeCun et al. 2010). An empirical comparison is made between original ELM and RBM-ELM.

By experimenting such approach, we mainly focus on the following aspects: (1) significance of model performance; (2) influence of the hidden layer size; (3) variation of parameters during training. And finally, we summarize the effect of RBM unsupervised pre-training on ELM.

Algorithm 1: Proposed RBM-ELM Algorithm

Input: ;
Training sample $\mathbf{S} = (\mathbf{X}, \mathbf{T})$;
Activation function $g(\cdot)$;
Number of hidden nodes m;
Number of iteration ITER;
Learning rate η

Output: Esimated $\mathbf{w}^*$, $\mathbf{a}^*$, $\mathbf{b}^*$ and output weight $\hat{\boldsymbol{\beta}}$

Randomly assign input weight $\mathbf{w}^1$ and bias $\mathbf{a}^1$, $\mathbf{b}^1$;
for $t = 1$ *to* ITER **do**
  Compute the gradient $(\Delta\mathbf{w}^t, \Delta\mathbf{a}^t, \Delta\mathbf{b}^t)$ by (9);
  Update $(\mathbf{w}^{t+1}, \mathbf{a}^{t+1}, \mathbf{b}^{t+1})$ by

$$\begin{cases} \mathbf{w}^{t+1} = \mathbf{w}^t + \eta \frac{\Delta \mathbf{w}^t}{N} \\ \mathbf{a}^{t+1} = \mathbf{a}^t + \eta \frac{\Delta \mathbf{a}^t}{N} \\ \mathbf{b}^{t+1} = \mathbf{b}^t + \eta \frac{\Delta \mathbf{b}^t}{N} \end{cases} \tag{16}$$

end
Let $\mathbf{w}^* = \mathbf{w}^{\mathrm{ITER}+1}$, $\mathbf{a}^* = \mathbf{a}^{\mathrm{ITER}+1}$, $\mathbf{b}^* = \mathbf{b}^{\mathrm{ITER}+1}$;
Compute the hidden layer output matrix $\mathbf{H} = g(\mathbf{w}^*\mathbf{X} + \mathbf{b}^*)$;
Compute the output weight $\hat{\boldsymbol{\beta}} = \mathbf{H}^{\dagger}\mathbf{T}$;
return $(\mathbf{w}^*, \mathbf{a}^*, \mathbf{b}^*, \hat{\boldsymbol{\beta}})$

4.1 Datasets and experiment

We collect 14 representative classification datasets which focus on various learning fields. These datasets contain both categorical attributes and numerical attributes, which will be used to evaluate the performance of RBM-ELM. The metadata of these datasets with training configurations are shown in Table 1.

Datasets 1–5 are of categorical attributes and datasets 6–14 are of numerical attributes. All numerical attributes are normalized within the range of [0, 1]. In order to handle categorical data, we need to transfer each category into a list of numbers. For example, suppose that the categorical attribute $F = \{x, y, z\}$. We transform F into 3 attributes by taking value either 0 or 1, which are indicators of x, y, z. Since one categorical attribute will be extended to a list of indicators, the final dataset feeds to the model may have an obvious increase in attribute number.

Both ELM and RBM-ELM are trained according to the parameters column in Table 1. We apply the same number of hidden nodes m in hidden layer on two models. For RBM-ELM, η is the leaning rate and ITER is the maximum iteration in Algorithm 1. To comprehensively verify the model generalization ability and avoid over-fitting, the tenfold cross-validation is applied on each dataset. During training, the original dataset is randomly partitioned into 10 equal sized subsets. Of the 10 subsets, a single subset is retained as the validation set for testing the model, and the remaining 9 subsets combined are used as training samples. The cross-validation process is then repeated 10 times with each of the 10 subsets used exactly once as the

Table 1 Datasets for performance comparison

ID	Dataset	#Example	#Attribute	#Class	Parameters
1	MNIST	55,000	748	10	$m = 800;\ \eta = 0.01;\ \text{ITER} = 30$
2	Connect-4	67,557	42(126)	3	$m = 500;\ \eta = 0.01;\ \text{ITER} = 10$
3	Chess	3196	36	2	$m = 200;\ \eta = 0.01;\ \text{ITER} = 30$
4	Nursery	10,368	8(27)	5	$m = 300;\ \eta = 0.01;\ \text{ITER} = 30$
5	HIV-1 Protease Cleavage	6590	8(160)	2	$m = 220;\ \eta = 0.01;\ \text{ITER} = 30$
6	Pen-Based Recognition	10,992	16	10	$m = 50;\ \eta = 0.01;\ \text{ITER} = 20$
7	EEG Eye State	14,980	15	2	$m = 100;\ \eta = 0.01;\ \text{ITER} = 30$
8	MAGIC Gamma Telescope	19,020	11	2	$m = 100;\ \eta = 0.01;\ \text{ITER} = 30$
9	ISOLET	7797	618	26	$m = 800;\ \eta = 0.01;\ \text{ITER} = 30$
10	Shuttle	58,000	9	7	$m = 100;\ \eta = 0.01;\ \text{ITER} = 50$
11	Letter Recognition	20,000	16	26	$m = 350;\ \eta = 0.01;\ \text{ITER} = 30$
12	Image Segmentation	2310	19	7	$m = 200;\ \eta = 0.01;\ \text{ITER} = 10$
13	Sensorless Drive	58,509	49	11	$m = 200;\ \eta = 0.01;\ \text{ITER} = 30$
14	MiniBooNE Particle	130,065	50	2	$m = 300;\ \eta = 0.01;\ \text{ITER} = 30$

Number in bracket is the number of numerical attributes after transformation

Table 2 Tenfold cross-validation of selected datasets

ID	ELM		RBM-ELM		RBM-ELM versus ELM
	Accuracy (%)	Avg. training time (s)	Accuracy (%)	Avg. training time (s)	p value
1	89.74 ± 0.0120	4.81	$\mathbf{94.38 \pm 0.0125}$	80.71	< 0.001
2	76.38 ± 0.0043	16.82	$\mathbf{78.16 \pm 0.0037}$	84.15	< 0.001
3	94.83 ± 0.0161	0.29	$\mathbf{98.46 \pm 0.0058}$	5.00	< 0.001
4	92.31 ± 0.0092	1.88	$\mathbf{95.43 \pm 0.0053}$	28.50	< 0.001
5	90.92 ± 0.0052	0.54	$\mathbf{92.42 \pm 0.0080}$	13.69	< 0.001
6	93.53 ± 0.0093	0.18	$\mathbf{95.63 \pm 0.0079}$	2.48	< 0.001
7	$\mathbf{76.90 \pm 0.0105}$	0.37	72.44 ± 0.0177	11.41	< 0.001
8	85.60 ± 0.0084	0.46	86.06 ± 0.0110	12.79	0.3329
9	90.38 ± 0.0096	5.69	$\mathbf{94.21 \pm 0.0075}$	41.05	< 0.001
10	$\mathbf{99.19 \pm 0.0019}$	1.86	95.33 ± 0.0272	46.24	< 0.001
11	$\mathbf{86.96 \pm 0.0047}$	3.50	85.59 ± 0.0045	32.99	0.0031
12	$\mathbf{95.71 \pm 0.0127}$	0.23	93.20 ± 0.0096	1.32	< 0.001
13	$\mathbf{84.36 \pm 0.0059}$	5.37	77.11 ± 0.0054	82.81	< 0.001
14	91.60 ± 0.0019	16.03	$\mathbf{92.47 \pm 0.0014}$	348.85	< 0.001

For each dataset, the highest training accuracy is in bold face

validation set. The mean and standard deviation of testing accuracy on validation set are calculated for performance comparison. The algorithms are implemented in Python3.5 under the hardware environment with AMD Ryzen7 1700X CPU, 32GB RAM and 64-bit Ubuntu 17.04LTS operating system.

4.2 Result analysis

The experimental results are listed in Table 2. Highest testing accuracy is in bold face, and the Student's t test is given to show the significance between two models. From the table, we can view that the testing accuracy of RBM-ELM is better than ELM on 8 out of 14 datasets. And the small $pvalue$ indicates that the difference between two models is significant. Then, the situation reversed for the other 5 datasets, such that ELM has the better performance. In Sect. 4.3, we will show that this low accuracy on RBM-ELM is caused by the discontinuity of generalized inverse proved in Sect. 3. Furthermore, for dataset ID=8, the two algorithms are giving almost the same performance. It is hard to say which one is uniformly better for ELM and RBM-ELM with respect to the testing

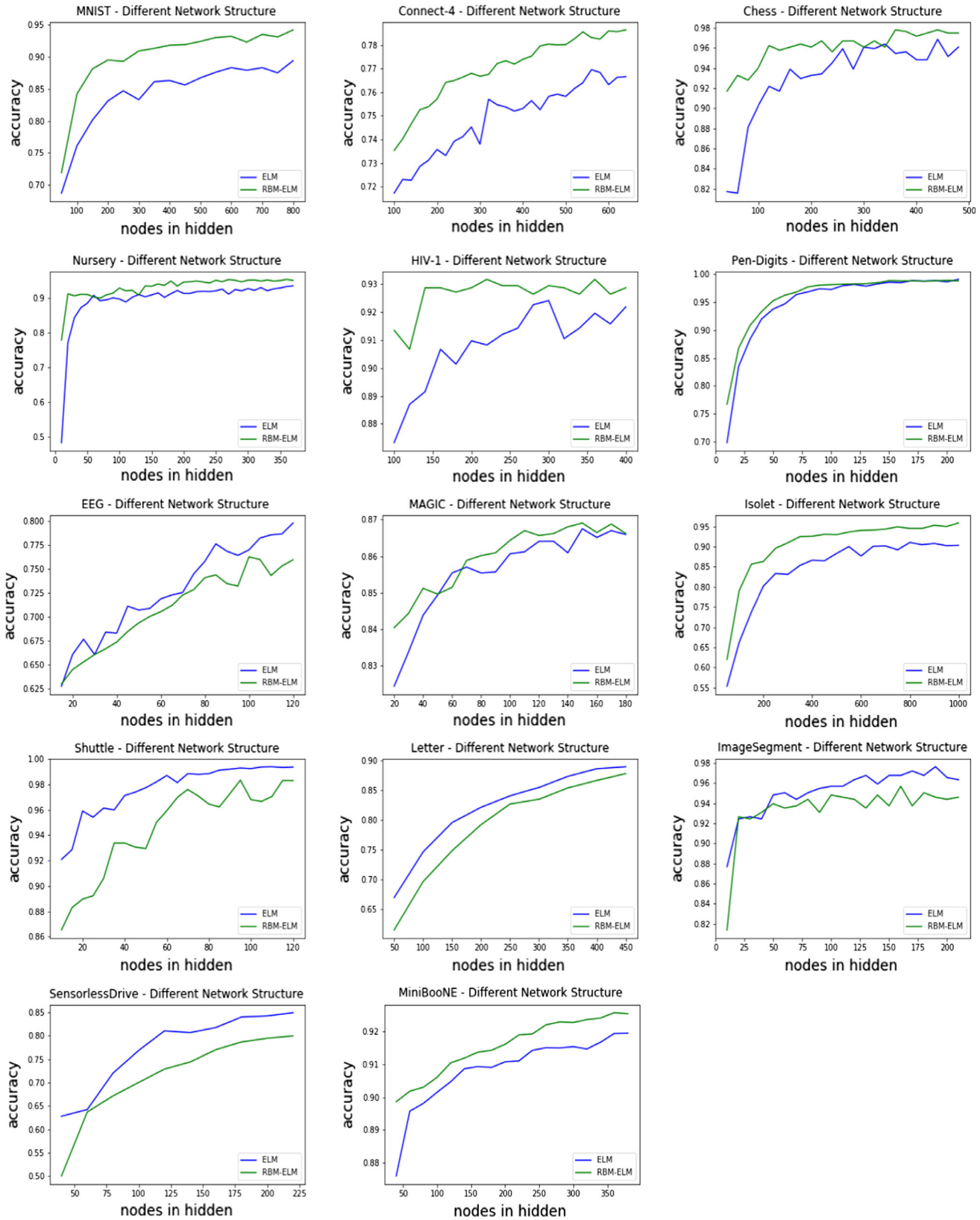

Fig. 3 Different network structures

accuracy. In another word, the effect of RBM pre-training depends on the datasets. Noting that the above experiments are conducted with fixed size of hidden layer. The hidden layer has the same number of nodes for the two models. Whether the change of hidden layer size would have a critical impact on the testing accuracy is still unknown. Especially, we are concerning whether the performance advantage will remain when number of hidden nodes changes. To answer this question, we conduct an additional experiment. For each dataset, 90% of the samples are selected as the training set and the remaining 10% as the testing set. Starting close to the number of input attributes, let the number of hidden layer nodes increase in a certain step, we recorded the different testing accuracy.

From Fig. 3, each dataset in Table 1 is trained with different hidden structures. Considering the overall result, the accuracy increases with more hidden nodes, and the performance advantage holds when changing the hidden layer size. For example, the first sub-figure is the MNIST handwritten digits dataset. With hidden layer varying from 50 to 800 with step 50, the RBM-ELM testing accuracy (green curve) is always above the ELM testing accuracy (blue curve) which matches the significance result in Table 2.

4.3 Network comparison

According to the previous results, we already find that there is a significant performance difference between RBM-ELM and ELM in most of the cases. By embedding hidden nodes designed using RBM, it is worthy to track value of parameters and intermediate results such as hidden matrix and its generalized inverse. Looking into such details, we can tell that the difference between two algorithms is not only from the final performance but also inside the training process. Thus, we carry out another experiment comparing the statistics of the weight matrix $\mathbf{W}$, hidden layer matrix $\mathbf{H}$ and its generalized inverse $\mathbf{H}^{\dagger}$ particularly on the datasets which RBM-ELM performs worse than ELM.

For the sake of simplicity, the visualization in this part is conducted only on a negative case (RBM-ELM performs worse than ELM), i.e., dataset Letter Recognition, and a positive case (RBM-ELM performs better than ELM), i.e., dataset MNIST. For other datasets, patterns are similar. With 800 hidden nodes, Figs. 4 and 5 show the distribution of weight variance for datasets Letter Recognition and MNIST, respectively. In the left two sub-figures, each dot stands for the variance of weights connecting to a hidden node. It is the column variance of $\mathbf{W}$. For example, input dataset has 16 attributes and hidden layer has 800 nodes, so hidden node h_1 has 16 connections which are $w_{1,1}$ to $w_{16,1}$, and the first dot represents the variance of these connections. It is surprisingly found from Fig. 4 that the RBM pre-trained weight matrix for dataset Letter Recognition has very low column variance, which means the weights are just slightly different from each other. As discussed in Sect. 2 and Algorithm 1, the weight matrix between visible and hidden layers will transform input data into hidden layer matrix. Therefore, with the low-variance weight matrix, the unlikely trends of hidden layer output could be expected. As a result, RBM-ELM performs worse than traditional ELM on dataset Letter Recognition. On the contrary, as shown in Fig. 5, the low-variance phenomenon does not exist for dataset MINST; thus, RBM-ELM performs better than traditional ELM on this dataset.

Figure 6 takes a random observation on the hidden layer output values for dataset Letter Recognition. Although both hidden layer matrices are full rank, the ELM hidden layer output is close to a uniform distribution within [0, 1], and while the RBM hidden layer output has low variance, it is nearly a perturbation around constant 0.5. While in Fig. 7 we repeat the procedure in MNIST dataset, there is no waning rank phenomenon in hidden layer output.

Furthermore, the next step is the same for both RBM-ELM and ELM: compute the generalized inverse for hidden layer matrix $\mathbf{H}$. Table 3 reveals the difference of generalized inverse (GI) matrix between ELM and RBM-ELM for dataset

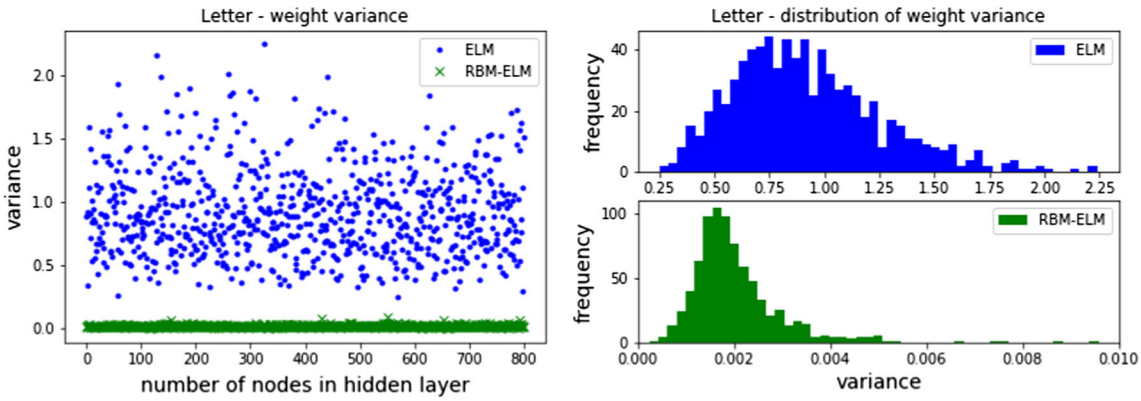

Fig. 4 Weight variance comparison between ELM and RBM-ELM on Letter

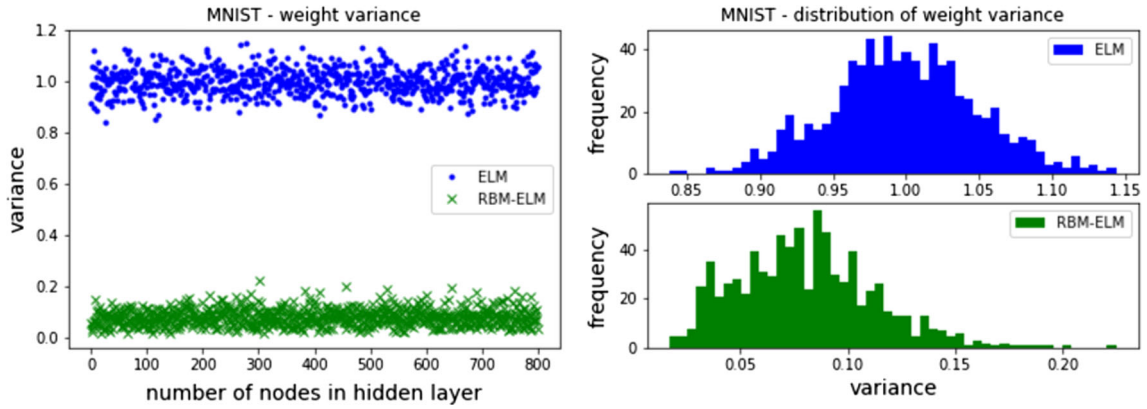

Fig. 5 Weight variance comparison between ELM and RBM-ELM on MNIST

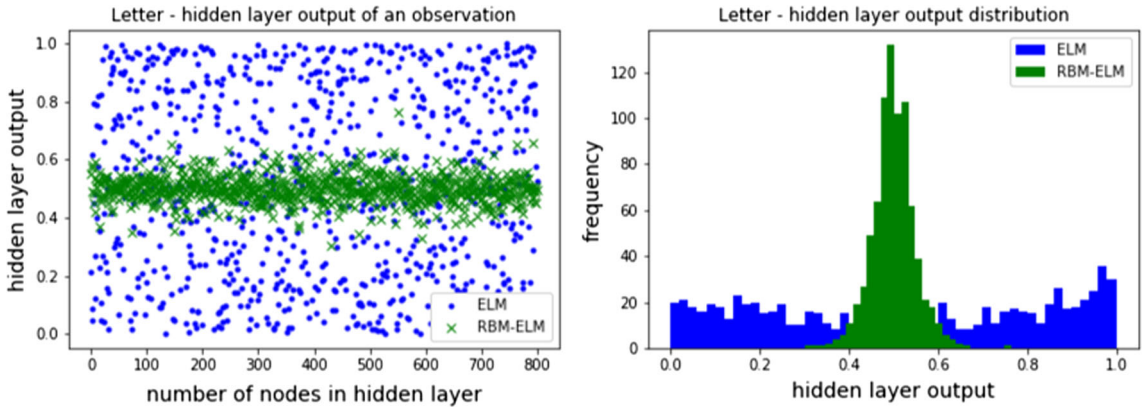

Fig. 6 Hidden layer output comparison between ELM and RBM-ELM on Letter

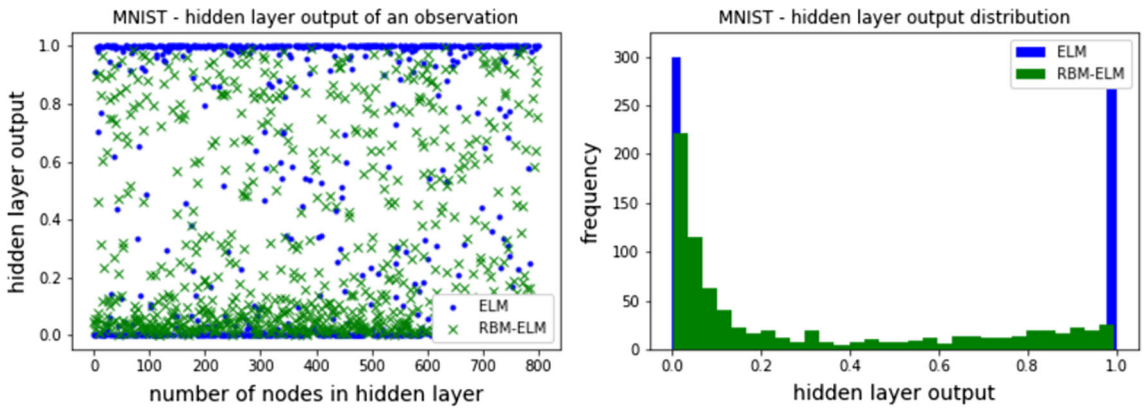

Fig. 7 Hidden layer output comparison between ELM and RBM-ELM on MNIST

Letter Recognition. Both have zero mean, and the large variance of GI in RBM-ELM is noteworthy. This phenomenon is actually due to the discontinuity of generalized inverse in waning rank matrix. The experimental result matches the theoretical analysis in Sect. 3.

5 Conclusion

In this paper, to study how initial weights affect the non-iterative training, the RBM is used as an unsupervised pre-training phase for ELM. We first give a detailed illustration

Table 3 Generalized inverse (GI) comparison

Model	GI mean	GI variance
ELM	1.7052×10^{-7}	0.0362
RBM-ELM	1.5632×10^{-7}	42052.53

about background and related works followed by introducing the RBM-ELM algorithm. Then, we prove the discontinuity of generalized inverse if the full rank matrix is close to a perturbation from waning rank. Next, to evaluate the model, some experiments were carried out with benchmark datasets and comparison was made between standard ELM and RBM-ELM. With the empirical study, we summarize the effect of RBM combined with ELM. The conclusions can be listed as follows:

1. For some of the experimental datasets, the performance of unsupervised pre-training using RBM is significantly better than random initiating.
2. Increasing hidden layer size by adding more nodes, the testing accuracy is improved on both standard ELM and RBM-ELM. And for a particular dataset, the advantage of RBM-ELM remains when hidden layer size changes.
3. According to our experiments, the RBM-ELM is not always surpass the standard ELM. The training of RBM could lead to a low- variance scenario in hidden layer. Because of the discontinuity of generalized inverse, it will finally cause large parameter variation in solution.
4. When the hidden layer is floating around a constant, it should be treated as perturbation of waning rank matrix, instead of full rank.

Acknowledgements This work was supported in part by the National Natural Science Foundation of China (Grants 61772344 and 61732011), in part by the Natural Science Foundation of SZU (Grants 827-000140, 827-000230, and 2017060), in part by the Youth Foundation Project of Hebei Natural Science Foundation of China (F2018511002), in part by Macao Science and Technology Development Funds (100/2013/A3& 081/2015/A3), and in part by the Interdisciplinary Innovation Team of Shenzhen University.

Compliance with ethical standards

Conflict of interest All the authors declare that they have no conflict of interest.

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors.

Informed consent Informed consent was obtained from all individual participants included in the study.

References

Bengio Y, Lamblin P, Popovici D, Larochelle H (2007) Greedy layer-wise training of deep networks. In: Advances in neural information processing systems, pp 153–160

Chen L, Yang L, Sun C, Xi H (2017) A fast RBM-hidden-nodes based extreme learning machine. In: Control And Decision Conference (CCDC), 2017 29th Chinese, IEEE, pp 2121–2126

Ding L, Han B, Wang S, Li X, Song B (2017a) User-centered recommendation using us-elm based on dynamic graph model in e-commerce. Int J Mach Learn Cybernet. https://doi.org/10.1007/s13042-017-0751-z

Ding S, Zhang N, Zhang J, Xu X, Shi Z (2017b) Unsupervised extreme learning machine with representational features. Int J Mach Learn Cybernet 8(2):587–595

Erhan D, Manzagol PA, Bengio Y, Bengio S, Vincent P (2009) The difficulty of training deep architectures and the effect of unsupervised pre-training. In: Artificial Intelligence and Statistics, pp 153–160

Erhan D, Bengio Y, Courville A, Manzagol PA, Vincent P, Bengio S (2010) Why does unsupervised pre-training help deep learning? J Mach Learn Res 11:625–660

Fu AM, Wang XZ, He YL, Wang LS (2014) A study on residence error of training an extreme learning machine and its application to evolutionary algorithms. Neurocomputing 146:75–82

Hinton GE (2006) Training products of experts by minimizing contrastive divergence. Neural Comput 14(8):1771–1800

Hinton G (2010) A practical guide to training restricted boltzmann machines. Momentum 9(1):926

Hinton GE, Osindero S, Teh YW (2006) A fast learning algorithm for deep belief nets. Neural Comput 18(7):1527–1554

Hinton G, Deng L, Yu D, Dahl GE, Ar Mohamed, Jaitly N, Senior A, Vanhoucke V, Nguyen P, Sainath TN et al (2012) Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups. IEEE Signal Process Mag 29(6):82–97

Huang GB, Zhu QY, Siew CK (2004) Extreme learning machine: a new learning scheme of feedforward neural networks. In: 2004 IEEE international joint conference on neural networks, 2004. Proceedings. IEEE, vol 2, pp 985–990

Huang GB, Zhu QY, Siew CK (2006) Extreme learning machine: theory and applications. Neurocomputing 70(1):489–501

Huang GB, Zhou H, Ding X, Zhang R (2012) Extreme learning machine for regression and multiclass classification. IEEE Trans Syst Man Cybernet Part B (Cybernet) 42(2):513–529

LeCun Y, Cortes C, Burges CJ (2010) Mnist handwritten digit database. AT&T Labs Available http://yann.lecun.com/exdb/mnist/

Li F, Liu H, Xu X, Sun F (2017) Haptic recognition using hierarchical extreme learning machine with local-receptive-field. Int J Mach Learn Cybernet. https://doi.org/10.1007/s13042-017-0736-y

Lichman M (2013) UCI machine learning repository. http://archive.ics.uci.edu/ml

Mao W, Wang J, Xue Z (2017) An elm-based model with sparse-weighting strategy for sequential data imbalance problem. Int J Mach Learn Cybernet 8(4):1333–1345

Meng L, Ding S, Xue Y (2017) Research on denoising sparse autoencoder. Int J Mach Learn Cybernet 8(5):1719–1729

Pacheco A, Krohling R, da Silva C (2017) Restricted boltzmann machine to determine the input weights for extreme learning machines. arXiv preprint arXiv:1708.05376

Parlett BN (1998) The symmetric eigenvalue problem. SIAM, Philadelphia

Salakhutdinov R, Mnih A, Hinton G (2007) Restricted boltzmann machines for collaborative filtering. In: Proceedings of the 24th international conference on machine learning. ACM, pp 791–798

Smolensky P (1986) Information processing in dynamical systems: Foundations of harmony theory. Technical report, Colorado University at Boulder Department of Computer Science

Wang R, Kwong S, Wang X (2012) A study on random weights between input and hidden layers in extreme learning machine. Soft Comput 16(9):1465–1475

Wang R, He YL, Chow CY, Ou FF, Zhang J (2015) Learning elm-tree from big data based on uncertainty reduction. Fuzzy Sets Syst 258:79–100

Wang R, Chow CY, Lyu Y, Lee V, Kwong S, Li Y, Zeng J (2017a) Taxirec: recommending road clusters to taxi drivers using ranking-based extreme learning machines. IEEE Trans Knowl Data Eng. https://doi.org/10.1109/TKDE.2017.2772907

Wang R, Xie H, Feng J, Wang FL, Xu C (2017b) Multi-criteria decision making based architecture selection for single-hidden layer feedforward neural networks. Int J Mach Learn Cybernet. https://doi.org/10.1007/s13042-017-0746-9

Wang XZ, Zhang T, Wang R (2017c) Noniterative deep learning: incorporating restricted boltzmann machine into multilayer random weight neural networks. IEEE Trans Syst Man Cybernet Syst. https://doi.org/10.1109/TSMC.2017.2701419

Wang XZ, Wang R, Xu C (2018) Discovering the relationship between generalization and uncertainty by incorporating complexity of classification. IEEE Trans Cybernet 48(2):703–715

Yu D, Deng L, Dahl G (2010) Roles of pre-training and fine-tuning in context-dependent dbn-hmms for real-world speech recognition. In: Proceedings of NIPS workshop on deep learning and unsupervised feature learning

Zhai J, Zhang S, Wang C (2017) The classification of imbalanced large data sets based on mapreduce and ensemble of elm classifiers. Int J Mach Learn Cybernet 8(3):1009–1017

Zhang H, Zhang S, Yin Y, Chen X (2017) Prediction of the hot metal silicon content in blast furnace based on extreme learning machine. Int J Mach Learn Cybernet. https://doi.org/10.1007/s13042-017-0674-8