Neural Computing and Applications (2020) 32:12685–12696
https://doi.org/10.1007/s00521-020-04719-8

ORIGINAL ARTICLE

A study on the relationship between the rank of input data and the performance of random weight neural network

Weipeng Cao[1] · Lei Hu[1] · Jinzhu Gao[2] · Xizhao Wang[1] · Zhong Ming[1]

Received: 6 July 2019 / Accepted: 7 January 2020 / Published online: 22 January 2020

Abstract

Random feature mapping (RFM) is the core operation in the random weight neural network (RWNN). Its quality has a significant impact on the performance of a RWNN model. However, there has been no good way to evaluate the quality of RFM. In this paper, we introduce a new concept called dispersion degree of matrix information distribution (DDMID), which can be used to measure the quality of RFM. We used DDMID in our experiments to explain the relationship between the rank of input data and the performance of the RWNN model and got some interesting results. We found that: (1) when the rank of input data reaches a certain threshold, the model's performance increases with the increase in the rank; (2) the impact of the rank on the model performance is insensitive to the type of activation functions and the number of hidden nodes; (3) if the DDMID of an RFM matrix is very small, it implies that the first **k** singular values in the singular value matrix of the RFM matrix contain too much information, which usually has a negative impact on the final closed-form solution of the RWNN model. Besides, we verified the improvement effect of intrinsic plasticity (IP) algorithm on RFM by using DDMID. The experimental results showed that DDMID allows researchers evaluate the mapping quality of data features before model training, so as to predict the effect of data preprocessing or network initialization without model training. We believe that our findings could provide useful guidance when constructing and analyzing a RWNN model.

Keywords Random weight neural network · Random vector functional link network · Extreme learning machine

1 Introduction

Since Turing A. conceived and discussed the concept of intelligent machinery in the 1950s [31, 32], the randomness in intelligent machines has received widespread attention. One of the most important components of the intelligent machinery is "unorganized machines." Turing A. divided the unorganized machines into two categories: A-type and B-type. Although the architectures and parameters of the unorganized machines contain some random factors, Turing believed that they can be trained as universal machines to do complex tasks under certain conditional constraints and with sufficient empirical data. Later, Webster [37] gave a more comprehensive and clear analysis of the relationship between Turing's unorganized machines and artificial neural networks and proposed an alternative solution for the B-type unorganized machines. In 1958, Rosenblatt [25] proposed the basic unit of modern neural networks: perceptron. Perceptron contains four components in its network structure: input area, projection area, association area, and response area. They are still the most commonly used components in current neural networks. Besides, Rosenblatt F. discussed the information storage and organization method in artificial neural networks. He also mentioned that the initialization of neural

✉ Weipeng Cao
cweipeng@pacific.edu

Lei Hu
2172272115@email.szu.edu.cn

Jinzhu Gao
jgao@pacific.edu

Xizhao Wang
xizhaowang@ieee.org

Zhong Ming
mingz@szu.edu.cn

[1] College of Computer Science and Software Engineering, Shenzhen University, Shenzhen 518060, China

[2] School of Engineering and Computer Science, University of the Pacific, Stockton, CA 95211, USA

networks should be largely random, which is consistent with Turing's vision of unorganized machines.

Based on the above works, the prototype of the modern random weight neural network (RWNN) was proposed in the 1990s [24, 28]. RWNN is a special type of feedforward neural network. The training mechanism of RWNN is quite different from the traditional iterative-based neural networks such as the backpropagation (BP) algorithm [12, 19]. For example, in BP algorithm, one needs to use a gradient descent-like approach to iteratively fine-tune all the parameters until it reaches acceptable accuracy. This iterative training process is often very time-consuming, especially when there are more than one hidden layers. Different from traditional neural networks, in RWNN, the input weights (i.e., the weights between the input layer and hidden layer) and the biases of hidden nodes (i.e., the thresholds of hidden nodes) are generated randomly from a given range and remain same during the training process, while the output weights (i.e., the weights between the hidden layer and output layer) are obtained by solving a system of linear matrix equations. It is a non-iterative learning process, so RWNN can train very fast with acceptable accuracy in some cases.

At present, there are two main research directions in RWNN: random vector functional link network (RVFL) [24] and extreme learning machine (ELM) [15]. It is noted that the training mechanisms of RVFL and ELM are the same, but there are some differences in the details [3]. For example, RVFL and ELM have different network structures. In RVFL, there is a direct connection between the input layer and output layer, which cannot be found in ELM. These differences greatly impact their performances on some issues [42].

In recent years, RWNN has attracted wide attention and researchers have used it to get many interesting results [5, 6, 8, 14, 33, 35, 38–41]. Some notable works include: Huang et al. [14] proved that ELM has the universal approximation ability; in [8], Dai et al. used RVFL to analysis the Alzheimer's disease data and achieved the state-of-the-art accuracy at the time; Chen et al. [5] proposed a motor-cognitive analytic framework based on ELM and used it to infer the cognitive wellness from motor patterns; Yang et al. [38] developed a super-node-based ELM model, that is, each hidden node is a sub-network structure, and they later extended it to a model with multiple hidden layers and unified the representation learning framework with unsupervised and supervised learning [39]; Uzair et al. [33] proposed an ELM-based blind domain adaptation algorithm, which can train the model without target domain samples.

Although the research based on RWNN has made great progress, one of the key problems has not been completely solved, that is, the quality assurance of random feature mapping (RFM). RFM is a core operation of RWNN, which transforms input data from the original feature space into another feature space. Its name comes from the fact that the randomly generated weights are used in the process. The quality of RFM has a significant impact on the model's performance.

Some related research works have emerged in recent years [2, 10, 16, 20, 29, 36, 42, 44] and can be classified into two categories: internal optimization and external optimization. Internal optimization methods improve the quality of RFM by optimizing the quality of the random input weights and hidden biases. For example, Zhang et al. [42] and Li et al. [20] studied the relationship between the internal parameters of RVFL and the model's performance, respectively, and many useful suggestions for RVFL model construction were given in their works; in [2] and [29], the authors studied the impact of the probability distribution on RWNN model initialization and provided some interesting observations; Wang et al. [36] stated that adding orthogonal constraints to the input weights of ELM allows the model to have a better ability to preserve the sample's structure. In [4, 21, 22], the authors used a biologically inspired adaptation rule called intrinsic plasticity (IP) to enhance the quality of RFM. Specifically, they used the IP algorithm to learn a set of slopes and biases for hidden nodes to make the output of the hidden layer to approximate a specific distribution with high entropy. The authors claimed that this method can maximize the information transmission from the input layer to the hidden layer, thereby improving the quality of RFM.

External optimization methods improves the quality of RFM by optimizing the data preprocessing process. For example, Fu et al. [10] studied how the performance of the ELM model changes with respect to the rank of the RFM matrix, which gave a clue to analyze the relationship between the rank of the input matrix and the performance of the model. Previously, we studied the relationship between the rank of input data and the performance of the ELM model and made some interesting observations [44]. For example, we found that when the input data change from non-full rank to full rank, the performance of the ELM model would gradually improve. However, there is no theoretical explanation for this observation. Huang et al. [16] studied the relationship between the rank of RFM matrix and the continuity of generalized inverse. They found that the minimum eigenvalue of the RFM matrix would be very small if the input data were non-full rank. In fact, according to our experimental results, we found that the minimum eigenvalues of the RFM matrixes obtained from two input matrixes with different non-full ranks are often too small to be compared. Therefore, we still cannot explain our observations discussed in [44].

In this paper, we still focus on studying the relationship between the rank of input data and the performance of the RWNN model. However, different from the previous works, our work covers the two main variations of RWNN, i.e., RVFL and ELM, and explains the relationship from a new perspective. Specifically, we aim to explain:

(1) The relationship between the rank of input data and the RVFL model performance;
(2) The sensitivity of the relationship between the rank of input data and the RWNN model performance to different activation functions;
(3) Our observations from a new theoretical perspective.

To our best knowledge, there is no previous study on the relationship between the rank of input data and the performance of the RVFL model. Although ELM and RVFL have the same learning mechanism, their RFM matrixes are quite different because of the different network structures. For example, the RFM matrix of RVFL is the concatenation of the input matrix and the output matrix of the hidden layer, while the RFM matrix of ELM is purely the output matrix of the hidden layer. Therefore, it is necessary to study the relationship for the RVFL network. In addition, the sensitivity of the activation function and the number of hidden nodes have not been considered in the previous works.

In our experiments, eight benchmark functions [23, 44] are used to generate the eligible datasets. Specifically, each benchmark function is used to generate 100 sub-datasets whose ranks increase from non-full rank to full rank with a fixed step size. These datasets are used to explain (1) and (2). In addition, we study (2) by changing the activation function of the RWNN model and the number of hidden layer nodes. For (3), in order to explain what we find out in (1) and (2), we propose a new concept called DDMID (dispersion degree of matrix information distribution).

To summarize, the main contributions of this paper are as follows:

- The relationship between the rank of input data and the performance of the RVFL model is studied for the first time, and some useful guidelines are given to optimize the process of data preprocessing;
- The sensitivities of the relationship between the rank of input data and the performance of the RWNN model with respect to the activation function and the number of hidden nodes are studied for the first time;
- A new theoretical perspective is proposed to explain the relationship between the rank of input data and the performance of the RWNN model, which is also a new indicator for evaluating the quality of RFM.

The rest of this paper is organized as follows: Section 2 first gives a brief review to RWNN and singular value decomposition (SVD). Then, we introduce the concept of DDMID in Sect. 3. The details of experimental settings are given in Sect. 4, followed by Sect. 5 that presents the experimental results and analysis. Finally, conclusions and our future works are discussed in Sect. 6.

2 Related works

In this section, we briefly review random weight neural networks (RWNN) with a focus on the random vector functional link network (RVFL) as well as singular value decomposition (SVD).

2.1 Random weight neural networks (RWNN)

As mentioned in Sect. 1, the learning mechanism of RWNN is different from traditional BP-based neural networks. Here, we use RVFL as an example to explain the learning mechanism of RWNN. RVFL is a typical RWNN and was proposed by Pao's group in the 1990s [24]. Since then, researchers have proposed many advanced algorithms based on it and applied them widely [7, 27, 34, 40, 43]. A typical structure of RVFL with a single hidden layer is shown in Fig. 1.

In Fig. 1, the input weights ω and hidden biases b are generated randomly and fixed throughout the learning process, while the output weights β are obtained analytically.

Suppose that there is a dataset $D = \{(x_i, t_i)|x_i \in R^d, t_i \in R\}, i = 1, 2, \ldots, N$, the learning process of RVFL composes of three steps:

Step 1 the linear operation phase. The input data are linearly represented by ω and b, i.e., $\omega \cdot x + b$;

Step 2 the nonlinear mapping phase. The output in Step 1 is nonlinearly mapped into a new feature space by the activation function $g(\cdot)$ of the hidden layer, i.e., $g(\omega \cdot x + b)$;

Step 3 the least squares solution phase. The output weights β are calculated by solving a system of linear equations.

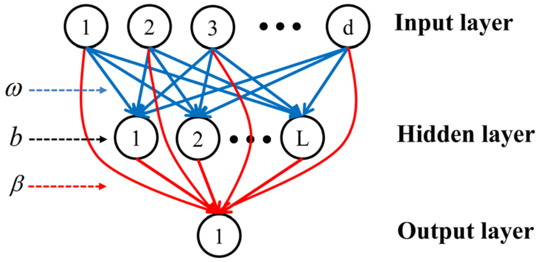


Fig. 1 The structure of RVFL with a single hidden layer

The process of RFM in RVFL consists of both Step 1 and Step 2, which can be represented as

$$\mathbf{H} = \begin{bmatrix} g(\omega_1 \cdot x_1 + b_1) & \cdots & g(\omega_L \cdot x_1 + b_L)x_1 \\ \vdots & \ddots & \vdots \\ g(\omega_1 \cdot x_N + b_1) & \cdots & g(\omega_L \cdot x_N + b_L)x_N \end{bmatrix}_{N \times (L+d)} \quad (1)$$

According to this analysis, we can see that the operations in Step 1 would not change the rank of input data. In Step 2, after the nonlinear mapping using the activation function of the hidden layer, the rank of the input data may be changed and can be full rank under certain conditions [10].

In RVFL, the RFM matrix **H** is the concatenation of the hidden layer output matrix and input matrix. Therefore, if the input matrix is non-full rank, **H** will also be a non-full-rank matrix with probability 1. This is different from ELM, in which the RFM matrix is full rank under certain conditions.

Once we have the RFM matrix **H**, Step 3 can be simplified as

$$\mathbf{H}\beta = \mathbf{T} \quad (2)$$

where $\beta = \begin{bmatrix} \beta_1 \\ \vdots \\ \beta_{L+d} \end{bmatrix}_{(L+d) \times 1}$, $\mathbf{T} = \begin{bmatrix} t_1 \\ \vdots \\ t_N \end{bmatrix}_{N \times 1}$

And the output weights β are computed by

$$\beta = \mathbf{H}^{+}\mathbf{T} \quad (3)$$

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

Now, the RVFL model can be represented as

$$\sum_{i=1}^{L} \beta_i g(\omega_i x + b_i) + \sum_{j=L+1}^{L+d} \beta_j x = y \quad (4)$$

Huang et al. [14] have proved that the solution obtained from (3) satisfies the following criterion

$$\min_{\|\beta\|} \left(\min \sum_{i=1}^{N} \|t_i - y_i\|^2 \right) \quad (5)$$

According to Bartlett's theory [1], this solution can minimize the structural error of the model, which implies that the model has a better generalization ability than those generated by other solutions.

2.2 Singular value decomposition (SVD)

Singular value decomposition (SVD) is an important matrix decomposition technique in linear algebra, which can be seen as the generalization of eigenvalue decomposition on arbitrary matrixes [18].

The process of SVD decomposes a linear transformation into three basic linear transformations, that is, rotation, scaling, and rotation again. Mathematically, given a matrix A with size $m \times n$, the SVD of A means that A can be represented by three special matrixes as shown below.

$$A = U\Sigma V^T \quad (6)$$

where U is a unitary matrix (i.e., $U^T U = I$) with size $m \times m$, Σ is the singular value matrix of A, and the size of Σ is $m \times n$. In Σ, except for the diagonal elements, the values of other elements are zero. The values of the diagonal elements are nonnegative real numbers, and they are also known as the singular values of A. V is also a unitary matrix (i.e., $V^T V = I$) with size $n \times n$. The columns of U and V are different orthogonal bases.

Further, the matrix A can be rewritten as

$$A = \sigma_1 \mu_1 v_1^T + \sigma_2 \mu_2 v_2^T + \ldots + \sigma_n \mu_n v_n^T = \sum_{i=1}^{n} \sigma_i \mu_i v_i^T = \sum_{i=1}^{n} M_i \quad (7)$$

where σ is the values of the diagonal elements in Σ and μ and v are the orthogonal bases in U and V, respectively.

As we know, σ_i is sorted in descending order in Σ, which implies the contribution of the corresponding item M_i to some extent. In other words, the larger σ_i, the greater the importance of corresponding M_i in the decomposition of A.

3 Dispersion degree of matrix information distribution (DDMID)

As mentioned in Sect. 1, the performance of the RWNN model largely depends on the quality of RFM. Since RFM matrix is transformed from the input data, to explain the relationship between the rank of input data and the model performance, intuitively one needs to study the influence of different ranks of input data on the RFM matrix. To achieve this goal, we introduce a new concept called dispersion degree of matrix information distribution (DDMID) in this section, which describes the characteristics of information distribution in a matrix. To better explain the concept of DDMID, we first introduce a preconception called the singularity of a matrix.

Definition 1 The singularity of a matrix. Suppose there are two matrixes, denoted as A and B, respectively. After performing SVD decomposition on A and B, respectively, if the information weight of the first k values in the singular value matrix Σ of A is larger than that of B, the singularity of A is considered larger than B.

For example, given two matrixes A and B with the same size 3 * 3 and the same rank 2:

$$A = \begin{pmatrix} 1 & 1 & 1 \\ 3 & 3 & 3 \\ 9 & 9 & 10 \end{pmatrix}, B = \begin{pmatrix} 1 & 1 & 1 \\ 3 & 3 & 3 \\ 1 & 5 & 9 \end{pmatrix} \tag{8}$$

The corresponding singular value matrixes of A and B after SVD decomposition are as follows:

$$\begin{aligned} \Sigma_A &= \begin{pmatrix} 17.09 & 0 & 0 \\ 0 & 0.26 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \\ \Sigma_B &= \begin{pmatrix} 11.38 & 0 & 0 \\ 0 & 2.72 & 0 \\ 0 & 0 & 0 \end{pmatrix} \end{aligned} \tag{9}$$

For the singular value matrix Σ_A of A, the weight of information stored in its first value is

$$17.09/(17.09 + 0.26) * 100\% = 98.50\% \tag{10}$$

For the singular value matrix Σ_B of B, the weight of information stored in its first value is

$$11.38/(11.38 + 2.72) * 100\% = 80.71\% \tag{11}$$

Obviously, the first value in Σ_A has a larger information weight than that of the first value in Σ_B. In other words, the information of Σ_A is more concentrated on the first position. In this case, we consider the singularity of A is large than B.

The larger the singularity of a matrix, the more the matrix information in a fewer number of singular values, and the smaller the information entropy of the matrix. In other words, the more singular a matrix is, the more linearly its rows (or columns) are related to each other, and the less information carried by the matrix.

As mentioned in Sec. 2, in Step 3 of modeling RWNN, that is, the least squares solution phase, the output weights are obtained analytically. SVD has been widely used to solve the pseudo-inverse of a matrix. Assume that a matrix A is decomposed by SVD to be:

$$A = U\Sigma V^T \tag{12}$$

Then, the pseudo-inverse of A is:

$$A^+ = V\Sigma^+ U^T \tag{13}$$

where Σ^+ is the pseudo-inverse of Σ.

This process can also be used to solve the least squares problem [11]. Because of this, we infer that the singularity of the RFM matrix may be intrinsically related to the generalization ability of the RWNN model.

We further infer that analyzing the characteristics of information distribution in the singular value matrix of the RFM matrix may give us some clues that can explain what we have observed in the experiments. To help us find these clues, we propose a new concept called dispersion degree of matrix information distribution (DDMID), which is used to describe the characteristics of information distribution in a matrix.

Definition 2 Dispersion degree of matrix information distribution (DDMID). Given a matrix **H**, perform SVD decomposition on **H** and obtain the corresponding singular value matrix Σ. Suppose the sum of the diagonal elements in Σ is S and the sum of the first k diagonal elements in Σ is S_k. Given a weight threshold p, if the proportion of S_k to S is greater than p, that is, $S_k/S \geq p$, the proportion of k to the number of the diagonal elements is called DDMID.

Taking the two matrixes in (8) as examples: Given $p = 90\%$, from (10) and (11), we know that the DDMIDs of the matrix A and B are as follows:

$$\begin{aligned} &\mathrm{DDMID_A} = 1/3, \\ &\mathrm{DDMID_B} = 2/3 \\ = \ &> \mathrm{DDMID_A} < \mathrm{DDMID_B} \end{aligned} \tag{14}$$

We can see that the dispersion degree of matrix information distribution of A is smaller than that of B, which means that the distribution of information in B is more uniform than A.

Next, we use DDMID as an indicator and take into account the relationship between the rank of input data and the performance of the RWNN model to study the influence of the rank of input data on the RFM matrix of RWNN.

4 Experimental settings

In this section, we explain the experimental settings in detail.

4.1 Datasets description

In our experiments, eight benchmark functions [23, 44] are chosen for generating eight types of datasets (denoted as **D1, D2, ..., D8**). The details of these eight benchmark functions are shown in Table 1. There are 2000 samples in total, and each sample has 100 attributes. The labels are generated by substituting the corresponding attributes' values in a specific benchmark function. For each type of datasets, there are 100 sub-datasets with ranks increasing gradually from non-full rank to full rank, i.e., from 1 to 100.

Table 1 Eight benchmark functions for constructing datasets

Name	Function	Initial range
Sphere	$f = \sum_{i=1}^{D} x_i^2$	$[-100, 100]^D$
Step	$f = \sum_{i=1}^{D} (\lfloor x_i + 0.5 \rfloor)^2$	$[-100, 100]^D$
Schwefel	$f = 418.9829D - \sum_{i=1}^{D} x_i \times \sin\sqrt{\lvert x_i \rvert}$	$[-500, 500]^D$
Rastrigin	$f = \sum_{i=1}^{D} \left[x_i^2 - 10\cos(2\pi x_i) + 10\right]$	$[-5.12, 5.12]^D$
Ackley	$f = -20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos 2\pi x_i\right) + 20 + e$	$[-32, 32]^D$
Rosenbrock	$f = \sum_{i=1}^{D-1}\left[100(x_{i+1} - x_i)^2 + (x_i - 1)^2\right]$	$[-30, 30]^D$
Schafferf	$f = \sum_{i=1}^{D-1}\left[\left(x_i^2 + x_{i+1}^2\right)^{0.25}\left(\sin\left(50\left(x_i^2 + x_{i+1}^2\right)^{0.1}\right)^2 + 1\right)\right]$	$[-100, 100]^D$
Quartic	$f = \sum_{i=1}^{D} i x_i^4 + random[0, 1)$	$[-1.28, 1.28]^D$

Here, we use the *Sphere* function as an example to show the process of generating datasets. The main steps of generating a dataset with the rank 80 are as follows.

Step 1 randomly generate a matrix with size 2000 * 80 using a *Uniform* distribution with the parameter $(-1, 1)$ and denote it as **D1-step1**. The rank of **D1-step1** is 80, the number of samples in **D1-step1** is 2000, and each sample has 80 attributes.

Step 2 for each sample, the values of 20 remaining attributes are obtained by linearly combining the values of existing 80 attributes with 20 different random coefficients. In this way, a new matrix with size 2000 * 100 is obtained, which is denoted as **D1-step2**. After that, **D1-step2** is normalized to meet the variables range requirements of the *Sphere* function.

Step 3 for each sample, substitute the attributes' values in the *Sphere* function and obtain the corresponding label. Then, the label vector is concatenated to **D1-step2** to obtain the final experimental dataset, which is denoted as **D1-80**. The size of **D1-80** is 2000 * 101.

Following these steps, each benchmark function in Table 1 can be used to generate 100 sub-datasets with ranks increasing gradually from non-full rank to full rank. We then use these datasets to study the relationship between the rank of input data and the performance of the RVFL model.

4.2 Parameters setting for RWNN

For RWNN, the parameters involved include the input weights, hidden biases, the number of hidden nodes, and the activation function. Our settings are as follows.

(1) Input weights and hidden biases. In our experiments, the input weights and hidden biases are randomly generated by the *Uniform* distribution on the intervals $(-1, 1)$ and (0, 1), respectively;

(2) The number of hidden nodes. To study the role of the number of hidden nodes in the relationship between the rank of input data and the performance of the model, we gradually increase the number of hidden nodes from 10 to 150 with the step size 10 and observe the changes in the relationship;

(3) The activation functions. Almost any nonlinear piecewise continuous functions can be used as the activation function in RWNN [14]. To study the sensitivity of the relationship between the rank of input data and the performance of the RWNN model to different activation functions, three commonly used activation functions in RWNN, that is, *Sigmoid* function, *RBF* (radial basis function), and *Sine* function, are used in our experiments. The mathematical expressions of these three activation functions are as follows:

Sigmoid: $G(\omega, x, b) = 1/(1 + \exp(-(\omega \cdot x + b)))$;

RBF: $G(\omega, x, b) = \exp\left(-b\|x - \omega\|^2\right)$;

Sine: $G(\omega, x, b) = \sin(\omega \cdot x + b)$.

4.3 Evaluation indicators

In our experiments, we use training and testing RMSEs (root mean square error), training and testing SDs (standard deviation) to evaluate the performance of the model. Training and testing RMSEs reflect the fitting ability of the model: the smaller the value, the better the model; training and testing SDs reflect the stability of the model: the smaller the value, the better the model. The mathematical expressions of RMSE and SD are as follows:

$$RMSE = \sqrt{\sum_{i=1}^{N} \frac{(y(i) - t(i))^2}{N}} \tag{15}$$

$$SD = \sqrt{\frac{\sum_{j=1}^{K} \left(e_j - \overline{e}\right)^2}{K - 1}} \tag{16}$$

where K is the number of independent experiments for each case, e is the error of the model for each experiment, and $\overline{e}$ is the mean value of the errors.

All the experiments in this paper are conducted using MATLAB R2016b on Windows 7, Intel(R) Core(TM) i7-6700 3.4 GHz CPU, and 32 GB RAM. The experimental result for each case is the average output of 50 trials.

5 Experimental results and analysis

In our experiments, we aim to answer the following questions:

1. With the same network parameters (i.e., the same number of hidden nodes and the same activation function), how does the RVFL model performance change with the increase in the rank of input data?
2. Does the activation function of RWNN (both RVFL and ELM) have an effect on the relationship between the rank of input data and the performance of the model?
3. When the number of hidden nodes changes, how does the relationship between the rank of input data and the performance of the RWNN model change?

5.1 Experimental results and analysis

For Question 1 and 2, experimental results are obtained and shown in Figs. 2 and 3. These two figures show the following:

(1) When the rank of input data reaches a certain value, the training and testing RMSEs and SDs of the RVFL model all decrease with the increase in the rank, which implies that the performance of the RVFL model improves with the increase in the rank and the performance reaches the best when the input data are full rank.

(2) This relationship between the rank of input data and the performance of the RVFL model is independent of the activation function.

(3) Compared with RVFL models using *RBF* or *Sine* functions as the activation function, models using *Sigmoid* function always achieve better performance.

To answer Question 3, we studied the effect of the number of hidden nodes in RVFL on the relationship between the rank of input data and the performance of the model. The experimental results are shown in Fig. 4. The figure shows the following:

(4) Although the prediction accuracies of RVFL models with the different number of hidden nodes are slightly different, the relationship between the rank of input data and the performance of the model is almost the same, which implies that the relationship is insensitive to the number of hidden nodes.

It is noted that we made the same observations described in (1)–(4) when using **D2-D8** datasets. Therefore, taking into account our earlier work using ELM as the base classifier to study the relationship between the rank of input data and the performance of the model [44], we come up with the following conclusion:

(5) When the rank of input data reaches a certain value, the performance of the RWNN model (both ELM and RVFL) improves with the increase in the rank. This relationship is insensitive to the number of hidden nodes and the type of activation functions.

This conclusion provides a valuable guidance for researchers to do better data preprocessing for the RWNN model.

5.2 Explanations for the experimental results

In our experiments, the DDMID value corresponding to each RFM matrix is calculated for the purpose of analysis. The weight threshold is set to 90% in our experiments. Specifically, each RFM matrix is first decomposed using the SVD technique, and then, the corresponding DDMID value of each RFM matrix can be calculated. Since the RFM matrix is transformed from the input data, a connection should exist between the rank of input data and the DDMID value of RFM matrix. Considering the relationship between the rank of input data and the performance of the RWNN model, we can also find out the relationship between the DDMID value and the performance of the RWNN model. Here, we use a simple example to illustrate the process of calculating the DDMID value of a RFM matrix.

Given a RFM matrix $\mathbf{H}$ with size $m \times n$, after performing SVD decomposition on $\mathbf{H}$, we can get its corresponding singular value matrix Σ_H. Suppose that the diagonal elements in Σ_H are denoted as $\{\lambda_1, \lambda_2, \cdots, \lambda_n\}$ and their sum is S. Given a weight threshold p (i.e., 90%). Starting from the first diagonal element in Σ_H, accumulate the diagonal elements one by one in order. Assume S_k is the sum of the first k diagonal elements. Once $S_k/S \geq p$, the

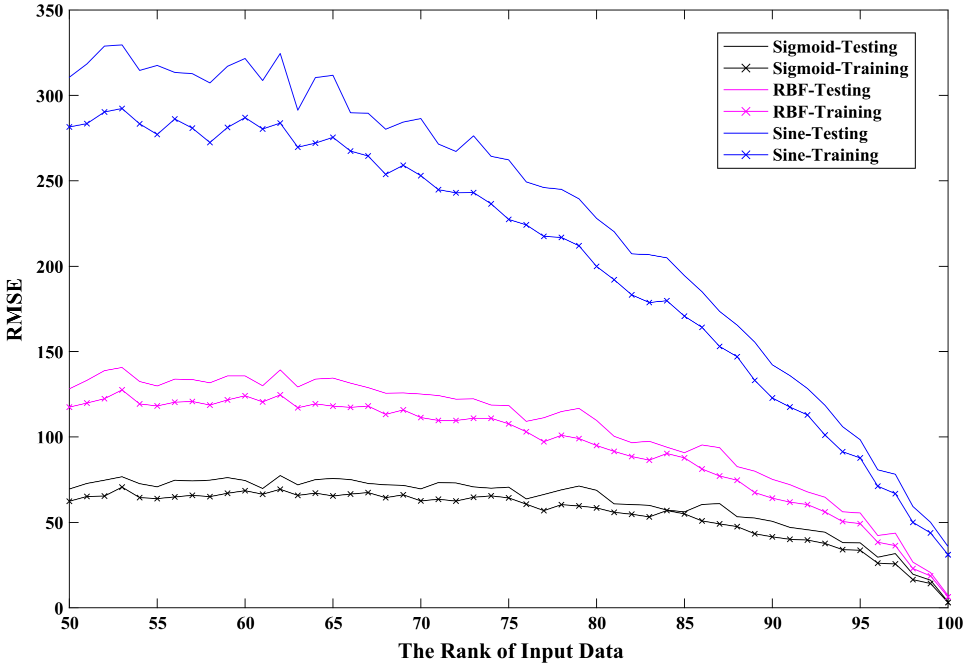


Fig. 2 Training and testing RMSE of RVFL model on **D1** (hidden nodes = 100)

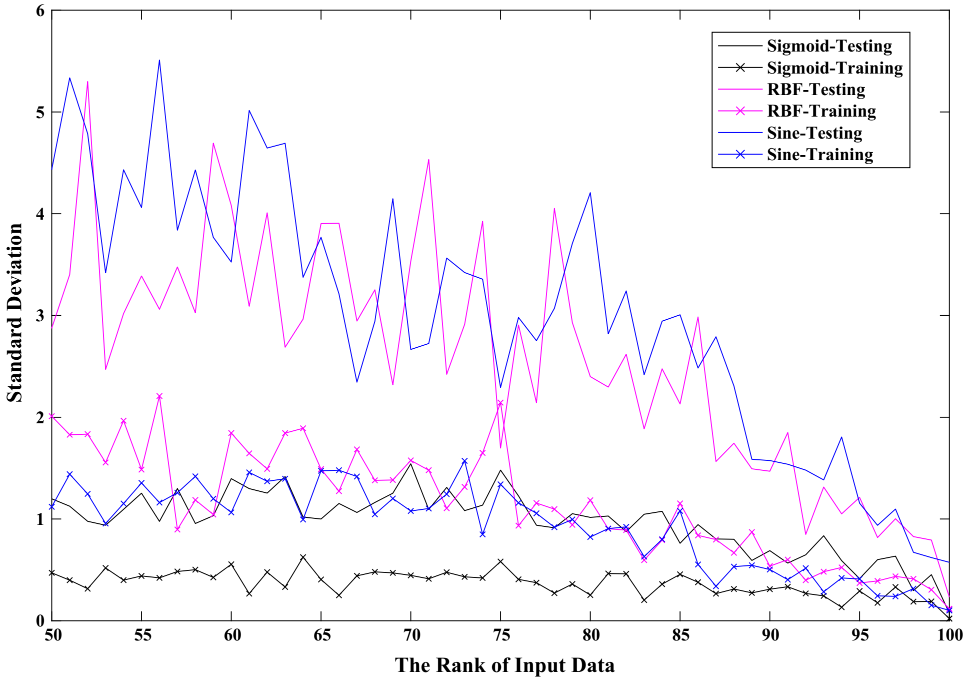


Fig. 3 Training and testing SD of RVFL model on **D1** (hidden nodes = 100)

accumulation process ends and the current k is recorded. The DDMID value of $\mathbf{H}$ is calculated by k/n.

In this way, we calculated the DDMID values of all RFM matrixes and found that the rank of input data has a significant impact on the DDMID of the corresponding RFM matrix. The experimental results from the dataset **D1** are shown in Fig. 5.

From Fig. 5, we see that as the rank of input data increases, the DDMID of the RFM matrix gradually increases, and this trend is independent of the number of hidden nodes. Considering this together with the experimental results shown in Figs. 2 and 3, we know that the testing RMSE and SD of the RVFL model gradually decrease as the rank of input data increases, which means that with the increase in DDMID, the generalization ability of the RVFL model increases. The experimental results from the datasets **D2-D8** also show similar trend. From these observations, we conclude:

Fig. 4 The effect of the number of hidden nodes on the relationship between the rank of input data and the testing RMSE of the model on **D1** (activation function: *Sigmoid* function)

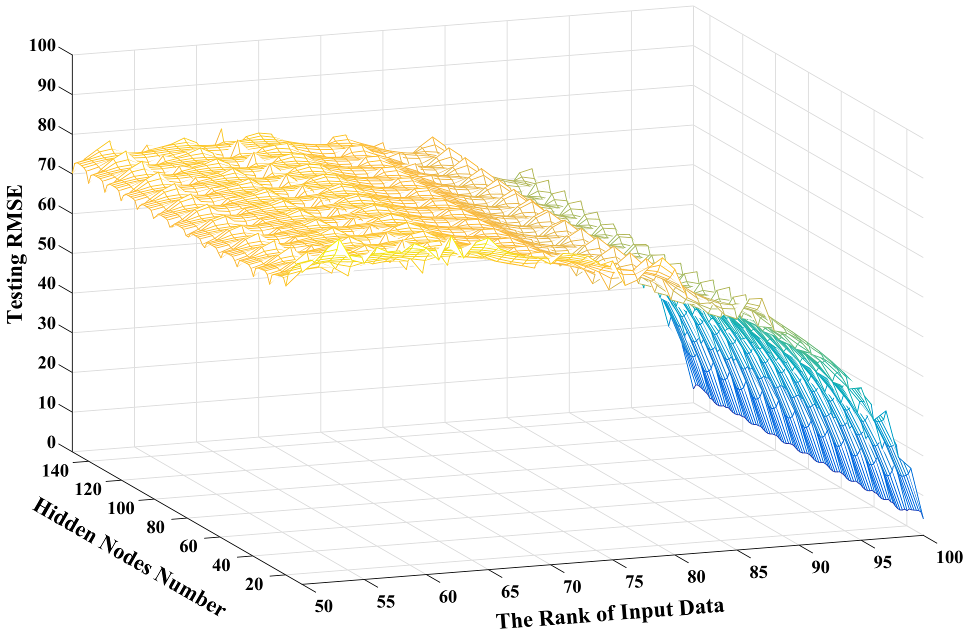
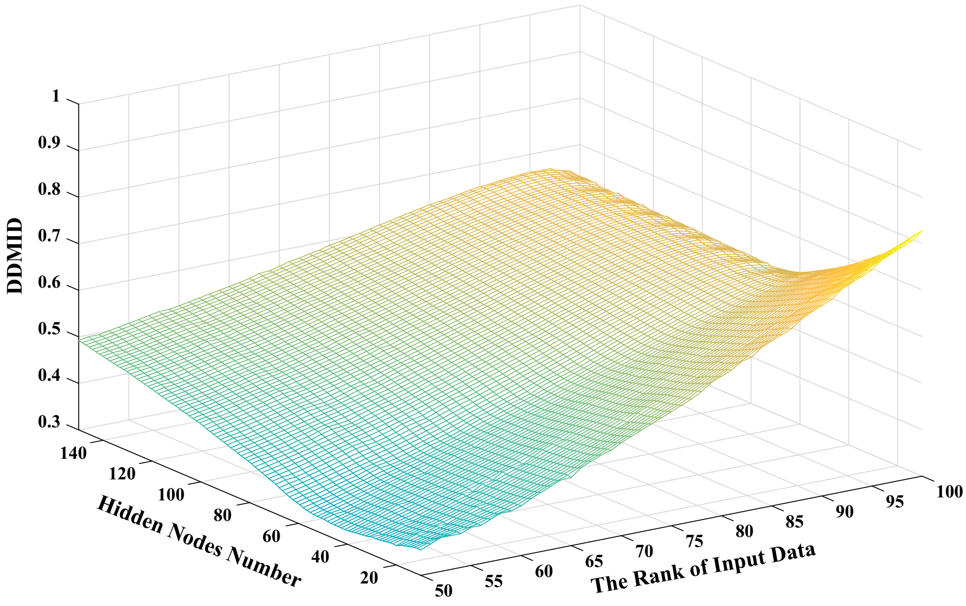


Fig. 5 The influence of the rank of input data and the number of hidden nodes on the DDMID of the RFM matrix (activation function: *Sigmoid* function)



(6) The DDMID can reflect the quality of the RFM matrix to some extent. Specifically, if the DDMID is small, it means that the first k singular values in the singular value matrix of the RFM matrix have too much weight, which implies that the quality of the RFM is not good and the final model may not achieve a good performance, and vice versa. Another explanation is that since the RFM in RWNN is followed by the least squares solution, the larger the DDMID, the more uniform the information distribution of the RFM matrix, and the better the solution.

Theoretical explanation of the above experimental phenomena is as follows: It can be known from (13) that the generalized inverse of the RFM matrix can be calculated by $H^{+} = V\Sigma^{+}U^{T}$. Actually, Σ^{+} is obtained by calculating the inverses of nonzero singular values in Σ and the singular values in Σ with values of zero remain unchanged. It can be inferred that if the nonzero singular values in Σ are too concentrated on the head position and occupy a large proportion of information, then the values on the head position of Σ^{+} will be very small because the inverses of these head elements are very small. Because the singular values in Σ with values of zero are still zero in Σ^{+}, there will be a large number of zeros or values close to zero in Σ^{+}. In this case, the continuity of H^{+} cannot be guaranteed [13], and then, it is difficult to obtain the optimal output weight by using $\beta = \mathbf{H}^{+}\mathbf{T}$.

DDMID reflects the dispersion degree of information distribution in the RFM matrix. If the value of DDMID of the RFM matrix is very small, it implies that only a few elements on the head position of the corresponding Σ occupy a large proportion of information. Furthermore, it can be inferred that the generalized inverse H^{+} corresponding to the RFM matrix is highly discontinuous with a large probability, which is not good for solving $\beta = \mathbf{H}^{+}\mathbf{T}$ [9]. Conversely, if the value of DDMID corresponding to the RFM matrix is very large, it implies that the generalized inverse H^{+} of the RFM matrix is continuous with a large probability, which is beneficial to the model to obtain a more stable solution.

Besides, we studied how the DDMID of the RFM matrix in ELM changes with the rank of input data and had a similar observation, that is, from a certain point, with the increase in the rank of input data, the DDMID of the RFM matrix gradually increases and the generalization ability of the model improves. To avoid repetition, the experimental details are not described here.

To further demonstrate the potential power of DDMID in real-life applications, we used DDMID to verify the improvement effect of intrinsic plasticity optimization method (IP) [4, 21, 22] on the quality of RFM. Specifically, we implemented IP algorithm and compared the performance of ELM with IP and ELM without IP on the dataset D1 and the DDMID values of the corresponding RFMs. The experimental results are shown in Table 2.

The experimental results in Table 2 show that using IP algorithm to add a specific constraint (i.e., a set of slopes and biases) to the input weights of ELM can improve the prediction performance of the model, which implies that IP algorithm improves the quality of the model's RFM. At the same time, we can also observe that the DDMID value of the RFM optimized by IP algorithm is higher than that without IP algorithm. This again shows that DDMID is an effective indicator to measure the quality of RFM.

Researchers can use the advantages of DDMID to evaluate the mapping quality of data features before model training, so as to predict the effect of data preprocessing or network initialization without model training.

Remark

(1) *The difference between DDMID and the condition number of the matrix* The condition number of the matrix can be used to measure the stability of a matrix. For example, for a linear system $\mathbf{H}\beta = \mathbf{T}$, if the condition number of H is very large (far greater than 1), a small change in T can cause a dramatic change in β, and the stability of the output result will be very poor. Conversely, if the condition number of H is very small (close to 1), the stability of the output result is relatively good. However, if the matrix H is not a full-rank matrix, the corresponding condition number will be infinite. For the RFM matrix in RVFL studied in this paper, it is a non-full-rank matrix in most cases (when the input matrix is not full rank). In this case, the stability of two non-full-rank RFM matrix cannot be compared by using the condition number of the matrix, and thus, it cannot be used to explain the phenomena in our study. However, the DDMID proposed in this paper can clearly reflect the influence of the rank of the input matrix on the quality of RFM (as shown in Fig. 5). Therefore, for this study, DDMID is more suitable than the conditional number of the matrix to explain the experimental phenomena.

(2) *The difference between our study and other works using SVD to analyze the hidden layer output matrix of neural networks, such as the work in* [30] **and** [26]. In [30], the authors noted that the singular values of the hidden layer output matrix could provide a quantification of the level of linear independency of the training patterns in hidden layer space. They found that small singular values might imply that the corresponding hidden layer nodes can be deleted, which does not seriously affect the performance of the model. Inspired by this observation, they proposed to use the singular values and the associate sensitivities from performing SVD on the hidden layer output matrix as indicators to select important neurons in the hidden layer. Similarly, in [26], the authors used principal component analysis (PCA) and SVD to decompose the matrixes of backpropagated errors and local gradient values associated with the hidden neurons in the training process of multi-layer perceptrons (MLP) and then pruned the hidden neurons corresponding to those

Table 2 Performance comparison of ELM with IP and ELM without IP on D1 (full-rank case)

Algorithm	DDMID	Training RMSE	Testing RMSE	Learning time(s)
ELM with IP	**0.79**	**3.12**	**3.23**	0.22
ELM without IP	0.75	3.38	3.55	**0.06**

relatively small singular values or eigenvalues. In this way, one can make the network structure of the model more compact without degrading the performance of the model too much.

The methods proposed in these two articles are mainly applied to pruning traditional neural networks (i.e., BP-based) to obtain a more compact network structure. In this study, we use SVD to analyze the information distribution of RWNN's RFM matrix and propose a new concept named DDMID to measure the quality of RFM. The aforementioned works and our work have significant differences in method design and application scenarios.

(3) *DDMID can also be used indirectly to assist in the design of new deep learning algorithms* Specifically, although the experiments in our paper are performed on RWNN with a single hidden layer, the conclusions of our study are also valid for deep RWNN. For example, for multi-layer ELM (ML-ELM) [17] (a typical deep learning framework in RWNN), all of its hidden layer parameters are obtained by pre-training using the ELM-based auto-encoder (ELM-AE). ELM-AE is an unsupervised feature extractor with only one hidden layer in its network structure. In this case, the effect of feature extraction of ELM-AE can be monitored by DDMID, and it is expected to obtain better features more efficiently, which will eventually help improve the prediction performance of the deep model.

6 Conclusions

In this paper, we studied the relationship between the rank of input data and the performance of the RWNN model, including both RVFL and ELM. We made the following interesting observations:

1. When the rank of input data reaches a certain threshold, the generalization ability of the RVFL model increases as the rank of input data increases and its performance reaches the best when the input matrix is full rank.
2. The observation in (1) is independent of the type of activation functions and the number of hidden nodes. In other words, different activation functions or the different number of hidden nodes may change the prediction accuracy of the model, but the relationship between its performance and the rank of input data is the same.

In addition, we introduced a new concept called dispersion degree of matrix information distribution (DDMID) and used it to explain what we have observed. Experimental results show that:

3. DDMID can reflect the quality of RFM (random feature mapping) of RWNN to some extent. Specifically, if the DDMID of an RFM matrix is very small, it means that the first k singular values in the singular value matrix of the RFM matrix contain too much information, which usually has a negative impact on the final solution of the RWNN model, and vice versa.
4. Furthermore, we used DDMID to verify the quality improvement of RFM by using the IP algorithm. The experimental results show that DDMID enables researchers to evaluate the mapping quality of data features accurately before model training and predict the effect of data preprocessing or network initialization without model training, which will greatly improve the efficiency of modeling.

These observations and analysis can be used as a guidance when constructing and analyzing RWNN models. However, there are still some unsolved issues in our work that we will be addressed in the future:

a. As mentioned above, we found that only when the rank of input data reaches a certain threshold, the relationship between the rank of input data and the performance of the RWNN model follows the trend as we observed. However, the threshold is not the same for different datasets and the relationship remains unclear when the rank of input data is below the threshold;

b. The weight threshold p in the DDMID was determined experimentally in the study. It is necessary to develop a simpler way to determine the value of p;

c. Adding specific noise to a non-full-rank input matrix can make it a full-rank matrix. In this case, how to evaluate the quality of the RFM remains to be studied.

Acknowledgements This work was supported in part by the National Natural Science Foundation of China (Grant 61672358 and Grant 61836005) and the Guangdong Science and Technology Department (Grant 2018B010107004)

Compliance with Ethical Standards

Conflict of interest The authors declared no potential conflict of interest with respect to the research, authorship, and/or publication of this article.

References

1. Bartlett PL (1998) The sample complexity of pattern classification with neural networks: the size of the weights is more important than the size of the network. IEEE Trans Inf Theory 44(2):525–536
2. Cao W, Gao J, Ming Z, Cai S, Zheng H (2017) Impact of probability distribution selection on RVFL performance. In: International conference on smart computing and communication. Springer, pp 114–124

3. Cao W, Wang X, Ming Z, Gao J (2018) A review on neural networks with random weights. Neurocomputing 275:278–287
4. Chen C, Jin X, Jiang B, Li L (2019) Optimizing extreme learning machine via generalized hebbian learning and intrinsic plasticity learning. Neural Process Lett 49(3):1593–1609
5. Chen Y, Hu C, Hu B, Hu L, Yu H, Miao C (2018) Inferring cognitive wellness from motor patterns. IEEE Trans Knowl Data Eng 30:2340
6. Chen Y, Song S, Li S, Yang L, Wu C (2018) Domain space transfer extreme learning machine for domain adaptation. IEEE Trans Cybern 49:1909
7. Cui W, Zhang L, Li B, Guo J, Meng W, Wang H, Xie L (2018) Received signal strength based indoor positioning using a random vector functional link network. IEEE Trans Ind Inform 14(5):1846–1855
8. Dai P, Gwadry-Sridhar F, Bauer M, Borrie M, Teng X (2017) Healthy cognitive aging: a hybrid random vector functional-link model for the analysis of alzheimer's disease. In: AAAI, pp 4567–4573
9. Fu A (2015) Study on the residence error, stability, and generalization capability of extreme learning machine. Ph.D. thesis, China Agricultural University
10. 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
11. Golub GH, Reinsch C (1970) Singular value decomposition and least squares solutions. Numer Math 14(5):403–420
12. Hecht-Nielsen R (1992) Theory of the backpropagation neural network. In: Wechsler H (ed) Neural networks for perception. Elsevier, Amsterdam, pp 65–93
13. Horn RA, Johnson CR (2012) Matrix analysis. Cambridge University Press, Cambridge
14. Huang GB, Chen L, Siew CK et al (2006) Universal approximation using incremental constructive feedforward networks with random hidden nodes. IEEE Trans Neural Netw 17(4):879–892
15. 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, vol 2. IEEE, pp 985–990
16. Huang Z, Wang X (2018) Sensitivity of data matrix rank in non-iterative training. Neurocomputing 313:386–391
17. Kasun LLC, Zhou H, Huang GB, Vong CM (2013) Representational learning with extreme learning machine for big data. IEEE Intell Syst 28(6):31–34
18. Laub AJ (1980) The singular value decomposition: its computation and some applications. IEEE Trans Autom Control 25(2):164–176
19. LeCun Y, Bengio Y, Hinton G (2015) Deep learning. Nature 521(7553):436
20. Li M, Wang D (2017) Insights into randomized algorithms for neural networks: practical issues and common pitfalls. Inf Sci 382:170–178
21. Neumann K, Emmerich C, Steil JJ (2012) Regularization by intrinsic plasticity and its synergies with recurrence for random projection methods. J Intell Learn Syst Appl 4(3):12
22. Neumann K, Steil JJ (2011) Batch intrinsic plasticity for extreme learning machines. In: International conference on artificial neural networks. Springer, pp 339–346
23. Ouyang H, Gao L, Li S, Kong X (2017) Improved global-best-guided particle swarm optimization with learning operation for global optimization problems. Appl Soft Comput 52:987–1008
24. Pao YH, Takefuji Y (1992) Functional-link net computing: theory, system architecture, and functionalities. Computer 25(5):76–79
25. Rosenblatt F (1958) The perceptron: a probabilistic model for information storage and organization in the brain. Psychol Rev 65(6):386
26. Santos JDA, Barreto GA, Medeiros CM (2010) Estimating the number of hidden neurons of the MLP using singular value decomposition and principal components analysis: a novel approach. In: 2010 Eleventh Brazilian symposium on neural networks. IEEE, pp 19–24
27. Scardapane S, Wang D, Uncini A (2018) Bayesian random vector functional-link networks for robust data modeling. IEEE Trans Cybern 48(7):2049–2059
28. Schmidt WF, Kraaijveld MA, Duin RP (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. IEEE, pp 1–4
29. Tao X, Zhou X, He YL, Ashfaq RAR (2016) Impact of variances of random weights and biases on extreme learning machine. JSW 11(5):440–454
30. Teoh EJ, Tan KC, Xiang C (2006) Estimating the number of hidden neurons in a feedforward network using the singular value decomposition. IEEE Trans Neural Netw 17(6):1623–1629
31. Cooper SB, Leeuwen JV (2013) Intelligent machinery. In: Alan turing his work and impact, pp 499–549
32. Turing AM (1996) Intelligent machinery, a heretical theory. Philos Math 4(3):256–260
33. Uzair M, Mian A (2017) Blind domain adaptation with augmented extreme learning machine features. IEEE Trans Cybern 47(3):651–660
34. Wang D, Li M (2017) Stochastic configuration networks: fundamentals and algorithms. IEEE Trans Cybern 47(10):3466–3479
35. Wang S, Deng C, Lin W, Huang GB, Zhao B (2017) Nmf-based image quality assessment using extreme learning machine. IEEE Trans Cybern 47(1):232–243
36. Wang W, Liu X (2017) The selection of input weights of extreme learning machine: a sample structure preserving point of view. Neurocomputing 261:28–36
37. Webster CS (2012) Alan turing's unorganized machines and artificial neural networks: his remarkable early work and future possibilities. Evol Intell 5(1):35–43
38. Yang Y, Wu QJ (2016) Extreme learning machine with subnetwork hidden nodes for regression and classification. IEEE Trans Cybern 46(12):2885–2898
39. Yang YM, Wu QJ (2016) Multilayer extreme learning machine with subnetwork nodes for representation learning. IEEE Trans Cybern 46(11):2570–2583
40. Ye H, Cao F, Wang D, Li H (2018) Building feedforward neural networks with random weights for large scale datasets. Expert Syst Appl 106:233–243
41. Zhang L, Deng P (2017) Abnormal odor detection in electronic nose via self-expression inspired extreme learning machine. IEEE Trans Syst Man Cybern Syst 99:1–11
42. Zhang L, Suganthan PN (2016) A comprehensive evaluation of random vector functional link networks. Inf Sci 367:1094–1105
43. Zhang L, Suganthan PN (2017) Visual tracking with convolutional random vector functional link network. IEEE Trans Cybern 47(10):3243–3253
44. Zhao X, Cao W, Zhu H, Ming Z, Ashfaq RAR (2018) An initial study on the rank of input matrix for extreme learning machine. Int J Mach Learn Cybern 9(5):867–879

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.