Research Article

BD-ELM: A Regularized Extreme Learning Machine Using Biased DropConnect and Biased Dropout

Jie Lai , Xiaodan Wang , Rui Li, Yafei Song , and Lei Lei

Air and Missile Defense College, Air Force Engineering University, Xi'an 710051, China

Correspondence should be addressed to Xiaodan Wang; afeu_wang@163.com

Received 2 July 2019; Accepted 1 February 2020; Published 29 February 2020

Academic Editor: Jefferson L. M. A. Gomes

Copyright © 2020 Jie Lai et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

In order to prevent the overfitting and improve the generalization performance of Extreme Learning Machine (ELM), a new regularization method, Biased DropConnect, and a new regularized ELM using the Biased DropConnect and Biased Dropout (BD-ELM) are both proposed in this paper. Like the Biased Dropout to hidden nodes, the Biased DropConnect can utilize the difference of connection weights to keep more information of network after dropping. The regular Dropout and DropConnect set the connection weights and output of the hidden layer to 0 with a single fixed probability. But the Biased DropConnect and Biased Dropout divide the connection weights and hidden nodes into high and low groups by threshold, and set different groups to 0 with different probabilities. Connection weights with high value and hidden nodes with a high-activated value, which make more contribution to network performance, will be kept by a lower drop probability, while the weights and hidden nodes with a low value will be given a higher drop probability to keep the drop probability of the whole network to a fixed constant. Using Biased DropConnect and Biased Dropout regularization, in BD-ELM, the sparsity of parameters is enhanced and the structural complexity is reduced. Experiments on various benchmark datasets show that Biased DropConnect and Biased Dropout can effectively address the overfitting, and BD-ELM can provide higher classification accuracy than ELM, R-ELM, and Drop-ELM.

1. Introduction

Extreme Learning Machine (ELM) [1], as the latest research achievement of Single-hidden Layer Feedforward Neural Networks (SLFNs), has attracted much attention due to its good generalization performance and fast training speed. Compared with traditional SLFNs, in ELM, the input weights and biases of the hidden layer are generated randomly without fine-tuning, and the hidden layer output weights are the global optimal solution solved by the least square method to avoid falling into the dilemma of local optimum [2, 3]. Although ELM has universal approximation [2, 4], it requires a considerable hidden nodes to ensure a good generalization performance, which is prone to overfitting. This makes the prevention of overfitting an urgent problem to ELM.

Regularization is one of the common methods to overcome the overfitting [5]. Various regularization methods are applied to the ELM algorithm to address the

overfitting [6–9]. Wang et al. [6] introduced L_1 -norm into ELM to make network parameters sparse and improve the generalization performance of ELM. A Regularized ELM (R-ELM) based on L_2 -norm loss was proposed by Deng et al. [7], which avoids overfitting and improves the robustness of the algorithm. By incorporating L_1 -norm loss with OP-ELM (Optimal Pruned ELM), Miche et al. [8] proposed the TROP-ELM (Tikhonov-regularized OP-ELM). TROP-ELM adopted both L_1 -norm and L_2 -norm regularization methods, and its generalization performance was significantly improved compared with ELM and OP-ELM.

Dropout [10] and DropConnect [11] are new regularization methods for fully connected networks. In Dropout and DropConnect, the output and connection weights of the hidden layer are, respectively, set to 0 with probability p , otherwise are kept with probability $(1 - p)$. Dropout and DropConnect are both very efficient ways of performing model averaging with neural networks [11]. In Dropout and DropConnect, each iteration is a training of a smaller

network, and the final network is the average of a large ensemble of networks which shared parameters. With the success of Dropout and DropConnect, their improved algorithms have been proposed by researchers [12–15]. Based on the difference of the importance of hidden nodes, Poernomo and Kang [12] proposed the Biased Dropout. In Biased Dropout, hidden nodes were divided into different groups according to their activation values, and different groups were dropped with different probabilities, respectively. Cao et al. [13], combining DropConnect with Adaptive Local Restricted Boltzman Machine (ALRBM), proposed the Adaptive Local Drop, in which the drop probability of connection weights changes with the distance function.

Inspired by Biased Dropout, this paper proposes the Biased DropConnect, which divides the connection weights into high and low groups by setting threshold and gives different drop probabilities. Then, applying the Biased DropConnect and Biased Dropout to the ELM, this paper proposes a Biased DropConnect and Biased Dropout based ELM (BD-ELM). The empirical studies show that Biased DropConnect and Biased Dropout can effectively overcome the overfitting of ELM, and compared with ELM, R-ELM, and Drop-ELM, BD-ELM can get higher classification accuracy on various benchmark datasets.

2. Related Work

2.1. Extreme Learning Machine. ELM is an efficient single-hidden layer feedforward neural network. Suppose the number of nodes in the input layer, hidden layer, and output layer is d , L , and m , respectively. For a given training samples $\{(\mathbf{x}_i, \mathbf{t}_i)\}_{i=1}^N$, where $\mathbf{x}_i = [x_{i1}, x_{i2}, \dots, x_{id}] \in \mathbf{R}^d$, the matrix expression of the network is as follows:

$$\mathbf{H}\boldsymbol{\beta} = \mathbf{T}, \quad (1)$$

where

$$\mathbf{H} = \begin{bmatrix} g(\mathbf{w}_1, b_1, \mathbf{x}_1) & \cdots & g(\mathbf{w}_L, b_L, \mathbf{x}_1) \\ \vdots & & \vdots \\ g(\mathbf{w}_1, b_1, \mathbf{x}_N) & \cdots & g(\mathbf{w}_L, b_L, \mathbf{x}_N) \end{bmatrix}_{N \times L}, \quad (2)$$

$$\boldsymbol{\beta} = \begin{bmatrix} \beta_1^T \\ \beta_2^T \\ \vdots \\ \beta_L^T \end{bmatrix}_{L \times m},$$

$$\mathbf{T} = \begin{bmatrix} \mathbf{t}_1^T \\ \mathbf{t}_2^T \\ \vdots \\ \mathbf{t}_L^T \end{bmatrix}_{N \times m},$$

where $\mathbf{w}_i = [w_{i1}, w_{i2}, \dots, w_{id}]^T$ is the input weights vector of i th hidden node, b_i is the bias, $\mathbf{w}_i$ and b_i are both generated randomly, $\boldsymbol{\beta}_i = [\beta_{i1}, \beta_{i2}, \dots, \beta_{im}]^T$ is the output weight vector, and $g(\cdot)$ is the activation function.

Training ELM is to find the least square solution of the output weights of the hidden layer. Without considering regularization, $\boldsymbol{\beta}$ can be calculated as follows:

$$\boldsymbol{\beta} = \mathbf{H}^\dagger \mathbf{T}, \quad (3)$$

where $\mathbf{H}^\dagger$ is the Moore–Penrose generalized inverse of $\mathbf{H}$. When $\mathbf{H}\mathbf{H}^T$ is nonsingular, $\mathbf{H}^\dagger = (\mathbf{H}^T \mathbf{H})^{-1} \mathbf{H}^T$; when $\mathbf{H}^T \mathbf{H}$ is nonsingular, $\mathbf{H}^\dagger = \mathbf{H}^T (\mathbf{H}^T \mathbf{H})^{-1}$.

2.2. Dropout and Biased Dropout. As a regularization method for fully connected networks, Dropout can effectively prevent overfitting [16]. During training, the output of the hidden nodes is set to zero with probability p , otherwise being kept with probability $(1 - p)$. The probability p is an important constant and need to be set in advance. When Dropout is applied to a fully connected hidden layer, the output is given as follows:

$$\mathbf{h} = \mathbf{M} \cdot g(\mathbf{W}\mathbf{v}), \quad (4)$$

where $(\cdot)$ denotes element-wise product, $\mathbf{v}_i = [\mathbf{v}_{i1}, \mathbf{v}_{i2}, \dots, \mathbf{v}_{in}]^T$ is the input of the hidden layer, $\mathbf{W}$ (of size $d \times n$) is the input weights (the bias is set to a fixed value of 1 and is included in $\mathbf{W}$ for simplicity), $g(\cdot)$ is the activation function, and $\mathbf{M} = [\mathbf{m}_1, \mathbf{m}_2, \dots, \mathbf{m}_d]^T$ is the binary mask matrix with each vector $\mathbf{m}_j \sim \text{Bernoulli}(1 - p)$ (all elements in vector are set to 1, or 0).

Most commonly used activation functions, such as relu, sigmoid, and tanh, have the property that $g(0) = 0$. Thus, (4) can be rewritten as follows:

$$\mathbf{h} = g(\mathbf{M} \cdot \mathbf{W}\mathbf{v}). \quad (5)$$

In this case, Dropout is applied at the input of the activation function.

Since each hidden node has different properties and different contribution to the network performance, using a universal probability for all nodes would render the effectiveness of Dropout itself [15]. But the Biased Dropout takes into account the difference of hidden nodes. Nodes with the high-activated value are more important, whose deletion will have a great impact on network performance. Therefore, nodes with high-activated value should have a lower drop probability. Viewed another way, limiting the drop probability for important nodes will make the network learn and converge faster than the regular Dropout [15].

In the Biased Dropout, the hidden nodes are divided into high and low group by the threshold of activation. The group with high activation nodes is assigned with a low drop probability to retain important information within, while another group with low activation nodes is given a high drop probability to keep the drop probability of the whole network to a fixed constant. Suppose the drop probability of the nodes with the high-activated value is p_D^h , the drop probability of the nodes with the low-activated value is p_D^l . The drop probability of network can be calculated as

$$p_D = \frac{(p_D^h \times n_h + p_D^l \times n_l)}{n}, \quad (6)$$

where n_h and n_l are the number of hidden nodes with the high-activated value and low-activated value, respectively, and n is the total number of hidden nodes. When the median of the activation value is chosen as threshold, $n_h \approx n_l$, $p_D = p_D^h + p_D^l/2$. The output of the hidden layer is calculated in the same way as (4), except the generation way of the mask matrix.

2.3. DropConnect. DropConnect, as a general regularization method of Dropout, sets the connection weights to 0 according to the drop probability p instead of the output of hidden nodes. Similar to Dropout, DropConnect also introduces dynamic sparsity in the network, but DropConnect is for connection weights, while Dropout is for the hidden output. When DropConnect is applied to a fully connected hidden layer, the output is given as

$$\mathbf{h} = g((\mathbf{M} \cdot \mathbf{W})\mathbf{v}), \quad (7)$$

where $\mathbf{M}$ (of size $d \times n$) is the mask matrix and each element $\mathbf{M}_{ij} \sim \text{Bernoulli}(1 - p)$. Comparing (7) with (4), because of different objects, the way of generating the mask matrix in Dropout and DropConnect is also different.

3. BD-ELM

3.1. Biased DropConnect. Inspired by the Biased Dropout, the Biased DropConnect was proposed. The basis of Biased Dropout is the difference of hidden nodes, while the basis of Biased DropConnect is the difference of connection weights. Different connection weights have different contributions to the network. Connection weights with the high value have greater impact on the network than those with the low value. The difference of connection weights has also been applied in an amplitude-based pruning technology by Han et al. [17], which can obtain a more compact network structure by setting threshold to prune lower weights.

In the Biased DropConnect, the connection weights are divided into two groups by threshold. High weight group is assigned with a low drop probability to hold more important information, while low weight group is given a high drop probability to keep the initial drop connect of the whole network. Assume that the drop probability of the high weight group is p_{DC}^h , and the drop probability of the low weight group is p_{DC}^l . Then, the drop probability of the network can be calculated as

$$p_{DC} = \frac{(p_{DC}^h \times m_h + p_{DC}^l \times m_l)}{m}, \quad (8)$$

where m_h and m_l are the number of connection weights with high value and low value, respectively, and m is the total number of connection weights. When the median weight is used as the threshold, $m_h \approx m_l$, the drop probability of network is $p_{DC} = p_{DC}^h + p_{DC}^l/2$. When using DropConnect, the calculation of the hidden output is same as (7), but the way of generating the mask matrix is different.

3.2. Biased DropConnect- and Biased DropConnect-Based Extreme Learning Machine. In order to prevent the

overfitting of ELM and improve its generalization performance, we incorporate Biased DropConnect and Biased Dropout regularization into ELM and propose BD-ELM. On the basis of ELM, BD-ELM regularizes the input weights of the hidden layer by Biased DropConnect and the output of the hidden layer by Biased Dropout. Then, it solves the output weights of the hidden layer and completes the training of the network. Compared with ELM, the input weights and output of the hidden layer in BD-ELM is sparser, which reduces network complexity and is conducive to improving the overfitting.

It should be noted that the application of Biased DropConnect and Biased Dropout to ELM requires additional setting of four parameters, p_{DC}^h , p_{DC}^l , p_D^h , and p_D^l . It is difficult to optimize those parameters. For reducing parameters, in BD-ELM, we set the drop probabilities of connection weights with the high value and hidden nodes with the high-activated value to 0, i.e., $p_{DC}^h = p_D^h = 0$. In other words, connection weights with a high value and hidden nodes with a high-activated value will be kept, and only the low will be dropped. Thus, the drop probabilities of Biased DropConnect p_{DC} and Biased Dropout p_D are only depended on the drop probabilities of connection weights with the low value p_{DC}^l and hidden nodes with the low-activated value p_D^l , respectively. When the medians of input weights and activation value of hidden nodes are chosen as thresholds, the drop probabilities of Biased DropConnect and Biased Dropout are $p_{DC} = p_{DC}^l/2$ and $p_D = p_D^l/2$, respectively. For the simplicity of comparison with other words, we still use p_{DC} and p_D as the measurement scales and input parameters.

Thus, the training of BD-ELM is as follows:

Input: training samples $\{\mathbf{X}, \mathbf{T}\} = \{\mathbf{x}_i, \mathbf{t}_i\}_{i=1}^N$, the number of hidden nodes L , the activation function $g(\cdot)$, the drop probability p_{DC} , p_D

Output: the output weights of hidden layer β

Step 1: randomly generate input weights $\mathbf{W}$ and bias $\mathbf{b}$

Step 2: generate the mask matrix $\mathbf{M}_{\text{weights}}$, according to $\mathbf{W}$ and the drop probability p_{DC}

Step 3: apply Biased DropConnect regularization to input weights, $\mathbf{W} = \mathbf{M}_{\text{weights}} \cdot \mathbf{W}$

Step 4: calculate the output of hidden layer $\mathbf{H}$

Step 5: generate the mask matrix $\mathbf{M}_{\text{output}}$, according to $\mathbf{H}$ and the drop probability p_D

Step 6: apply Biased Dropout regularization to output of hidden layer, $\mathbf{H} = \mathbf{M}_{\text{output}} \cdot \mathbf{H}$

Step 7: calculate the output weights of hidden layer β

4. Performance Evaluation of BD-ELM

All evaluations were carried out in Matlab 2017(b), running on a desktop with 4.2 GHz CPU, 16 GB RAM, and 1 TB hard disk. The following experiments were designed to evaluate the performance of BD-ELM (it is noted that all the results in this paper are averages of 30 repeated independent experiments):

Experiment 1: the influence of the number of hidden nodes on the performance of BD-ELM

Experiment 2: the influence of the drop probability on the performance of BD-ELM

Experiment 3: performance comparison with other algorithms

4.1. Data Description. In evaluations, Ionosphere, Diabetes, Vehicle, Image Segmentation, Satellite Image, and Letter Recognition in UCI datasets [18], as well as MNIST and Rectangles datasets, are used to train and test the BD-ELM. Details of datasets are shown in Table 1.

4.2. The Influence of the Number of Hidden Nodes. The number of hidden layer nodes is an important parameter of the neural network. Insufficient hidden nodes will result in underfitting, while excessive hidden nodes will lead to overfitting. ELM needs a large number of hidden nodes to ensure its generalization performance, but this is easy to lead to overfitting. To verify that Biased DropConnect and Biased Dropout can effectively prevent overfitting of ELM, we change the number of hidden nodes and observe the performance of ELM and BD-ELM on typical datasets.

The experimental datasets are Ionosphere, Diabetes, Vehicle, and Image Segmentation, and the range of the number of hidden nodes is $L = \{20, 40, \dots, 400\}$. In BD-ELM, the drop probabilities are $p_{DC} = p_D = p = \{0.2, 0.4\}$ and the thresholds are the medians of input weights and activation value of hidden nodes. And the activation function is sigmoid function. The variation of classification accuracy of ELM and BD-ELM with the number of hidden nodes is shown in Figures 1(a)–1(d).

As can be seen from Figure 1, BD-ELM is less prone to overfitting for the four datasets mentioned above. With the increase of the number of hidden nodes, serious overfitting appears in ELM. Compared with ELM, when overfitting appears, BD-ELM has more hidden nodes and less reduction in accuracy. This shows that Biased DropConnect and Biased Dropout can effectively improve the overfitting of ELM. The reason is that Biased DropConnect and Biased Dropout set the inputs weights and output of hidden nodes to 0, which is equivalent to removing these from the network. When the number of hidden is too large, setting part of input weights and output to 0 can remove the redundant information in the network, and the spare parameters reduce the network complexity so that BD-ELM can still keep a better accuracy than ELM.

And the generalization performance of BD-ELM with different drop probabilities show that a larger drop probability p is better for suppressing overfitting. But when p is too large, many input weights and output will be eliminated, and much important information will be lost, which will affect the generalization performance of BD-ELM. This is especially evident when hidden nodes are few. Therefore, it is very important to choose a suitable p for BD-ELM, according to the number of hidden nodes.

TABLE 1: The datasets used in the experiments.

Datasets	Attributes	Classes	Training data	Testing data
Ionosphere	34	2	180	171
Diabetes	8	2	390	378
Vehicle	18	4	446	400
Image segmentation	19	7	1210	1100
Satellite image	36	6	3217	3218
Letter recognition	16	26	16000	4000
MNIST	784	10	60000	10000
Rectangles	784	2	1200	50000

4.3. The Influence of the Drop Probability. The drop probability p , as the important parameter of Biased DropConnect and Biased Dropout, will affect the effectiveness of regularization and the generalization performance of BD-ELM. In order to clear the influence of the drop probability on BD-ELM, in this evaluation, a variable drop probability is adopted to observe the change of classification accuracy.

Assuming that the experimental data are Vehicle and Image Segmentation datasets, the number of hidden nodes is $L = 200$, the medians of input weights and output of hidden nodes are used as thresholds, the activation function is sigmoid function, and the ranges of drop probabilities of Biased DropConnect and Biased Dropout are $p_{DC} = p_1 = \{0, 0.05, 0.1, \dots, 0.5\}$ and $p_D = p_2 = \{0, 0.05, 0.1, \dots, 0.5\}$, respectively. The influence of drop probability on accuracy of BD-ELM is shown in Figures 2(a) and 2(b).

As shown in Figure 2, for Vehicle and Image Segmentation datasets, BD-ELM achieves optimal accuracy at $(p_{DC} = p_1, p_D = p_2) = (0.2, 0.3)$ and $(p_{DC} = p_1, p_D = p_2) = (0.3, 0.15)$, respectively.

As the drop probabilities p_{DC} and p_D increase, the accuracy of BD-ELM first rises and then decreases. This is because Biased DropConnect and Biased Dropout can remove more redundant information and enhance the positive influence of input weights with high value and hidden nodes with the high-activated value with the increase of p_{DC} and p_D . However, when the drop probability exceeds the critical point, a large number of connection weights or hidden nodes are dropped, and this means the great loss of important information, which makes the performance of BD-ELM not rise but decrease. Therefore, a suitable drop probability should be chosen after a concrete analysis, and it is roughly in $(0.1, 0.4)$.

4.4. Performance Comparison. For testing the comprehensive performance of BD-ELM, this evaluation will use all the datasets mentioned above to compare BD-ELM with ELM, R-ELM, and Drop-ELM [19]. Suppose the number of hidden nodes is in the range of $L = \{10, 20, \dots, 2000\}$, the regularization parameter in R-ELM is 10^4 , and the range of drop probability in Drop-ELM and BD-ELM is $p = \{0, 0.1, \dots, 0.5\}$. Multiple experiments have been performed by employing these parameters and the best performance is reported. The classification accuracy and training time of each algorithm are shown in Table 2 (the number in

(a)

(b)

(c)

(d)

FIGURE 1: The variation of classification accuracy of ELM and BD-ELM. The plots show that BD-ELM is less prone to overfitting than ELM. (a) Ionosphere, (b) Diabetes, (c) Vehicle, and (d) Image Segmentation.

(a)

(b)

FIGURE 2: The influence of drop probability on accuracy of BD-ELM. The plots show a suitable drop probability can improve the generalization performance of BD-ELM. (a) Vehicle and (b) Image Segmentation

TABLE 2: The classification accuracy (%) of ELM, R-ELM, Drop-ELM, and BD-ELM.

Datasets	Hidden nodes	ELM	R-ELM	Drop-ELM	BD-ELM
Ionosphere	40	76.67 $\pm$ 1.29	77.51 $\pm$ 0.72	76.98 $\pm$ 1.58	77.38 $\pm$ 1.32
Diabetes	60	84.21 $\pm$ 1.29	84.98 $\pm$ 1.35	85.04 $\pm$ 1.86	85.35 $\pm$ 1.32
Vehicle	140	80.40 $\pm$ 1.60	81.40 $\pm$ 1.16	81.65 $\pm$ 1.42	81.80 $\pm$ 1.50
Image segmentation	200	94.85 $\pm$ 0.34	95.40 $\pm$ 0.41	95.31 $\pm$ 0.52	95.90 $\pm$ 0.50
Satellite image	300	85.42 $\pm$ 0.24	86.20 $\pm$ 0.16	86.01 $\pm$ 0.29	86.34 $\pm$ 0.16
Letter recognition	600	76.97 $\pm$ 0.51	72.68 $\pm$ 0.57	77.22 $\pm$ 0.67	77.54 $\pm$ 0.66
MNIST	2000	94.78 $\pm$ 0.12	95.12 $\pm$ 0.15	94.89 $\pm$ 0.16	95.10 $\pm$ 0.19
Rectangles	2000	73.99 $\pm$ 0.66	73.85 $\pm$ 0.83	74.26 $\pm$ 1.29	74.52 $\pm$ 0.84

TABLE 3: The training time (10^{-2} s) of ELM, R-ELM, Drop-ELM, and BD-ELM.

Datasets	Hidden node	ELM	R-ELM	Drop-ELM	BD-ELM
Ionosphere	40	0.28	0.33	0.43	0.47
Diabetes	60	0.25	0.29	0.41	0.52
Vehicle	140	0.30	0.45	0.80	0.92
Image segmentation	200	0.49	0.65	1.26	1.71
Satellite image	300	0.83	1.12	2.43	2.75
Letter recognition	600	3.23	4.35	9.28	9.34
MNIST	2000	20.65	26.24	50.55	51.45
Rectangles	2000	0.84	1.24	1.87	2.00

boldface indicates the highest accuracy) and Table 3, respectively.

As can be seen from Table 3, in most datasets, BD-ELM can achieve higher classification accuracy, but requires more training time. (1) The performance of BD-ELM is better than that of ELM and R-ELM. Because, by dropping the unimportant weights and hidden nodes, BD-ELM has sparser parameter and less redundant information, which is conducive to improve the generalization performance, while the L_2 -norm in R-ELM does not have such ability [20]. (2) The performance of BD-ELM is better than that of Drop-ELM because BD-ELM takes into account the differences of connection weights and hidden nodes and preserves more important information contained in the weights with high value and hidden nodes with high-activated. But Drop-ELM only gives the same drop probability to all input weights and output of hidden nodes.

At the same time, BD-ELM needs to generate masking matrixes and this process is more complicated than that in Drop-ELM, thus its training time is more than other algorithms. These results show that it is necessary and effective to incorporate Biased DropConnect and Biased Dropout with ELM to prevent overfitting and improve generalization performance.

5. Conclusions

For the aim to address the overfitting and improve the generalization performance of ELM, this paper proposes a Biased DropConnect regularization method and applies the Biased DropConnect and Biased Dropout to the ELM to construct BD-ELM. BD-ELM divides the input weights and

output of hidden nodes into different groups by setting thresholds and gives different groups different drop probabilities, which enhance the sparsity and reduce complexity of the network. The empirical studies show that the Biased DropConnect and Biased Dropout can effectively prevent the overfitting, and BD-ELM can achieve better generalization performance on various benchmark datasets. However, BD-ELM requires more parameters and it is more difficult to find their best. These problems need further research and will be solved in the future. And inspired by other advanced machine learning methods [21, 22], the performance of ELM could be further improved.

Data Availability

The data used to support the finding of this study have been deposited in the GitHub repository (<https://github.com/LynnW0W/BD-ELM>).

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The research leading to these results has received funding from the National Natural Science Foundation of China (Grant nos. 61876189, 61273275, 61806219, and 61703426).

References

- [1] G. B. Huang, Q. Y. Zhu, and C. K. Siew, "Extreme learning machine: theory and applications," *Neurocomputing*, vol. 70, no. 1–3, pp. 489–501, 2006.
- [2] R. Zhang, Y. Lan, G. B. Huang et al., "Universal approximation of extreme learning machine with adaptive growth of hidden nodes," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 2, pp. 365–371, 2012.
- [3] G.-B. Huang, L. Chen, and C.-K. Siew, "Universal approximation using incremental constructive feedforward networks with random hidden nodes," *IEEE Transactions on Neural Networks*, vol. 17, no. 4, pp. 879–892, 2006.
- [4] G. B. Huang, H. Zhou, X. Ding et al., "Extreme learning machine for regression and multiclass classification," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 42, no. 2, pp. 513–529, 2012.
- [5] S. Chen, E. S. Chng, and K. Alkadhimi, "Regularized orthogonal least squares algorithm for constructing radial basis function networks," *International Journal of Control*, vol. 64, no. 5, pp. 829–837, 1996.

- [6] Y. Wang, D. Li, Y. Du, and Z. Pan, "Anomaly detection in traffic using L1-norm minimization extreme learning machine," *Neurocomputing*, vol. 149, no. 1, pp. 415–425, 2015.
- [7] W. Deng, Q. Zheng, and L. Chen, "Regularized extreme learning machine," in *Proceedings of the 2009 IEEE Symposium on Computational Intelligence and Data Mining*, pp. 389–395, IEEE, Nashville, TN, USA, March–April 2009.
- [8] Y. Miche, M. van Heeswijk, P. Bas, O. Simula, and A. Lendasse, "TROP-ELM: a double-regularized ELM using LARS and Tikhonov regularization," *Neurocomputing*, vol. 74, no. 16, pp. 2413–2421, 2011.
- [9] X. Luo, X. Chang, and X. Ban, "Regression and classification using extreme learning machine based on L1-norm and L2-norm," *Neurocomputing*, vol. 174, no. 1, pp. 179–186, 2016.
- [10] G. E. Hinton, N. Srivastava, A. Krizhevsky et al., "Improving neural networks by preventing co-adaptation of feature detectors," 2012, <http://arxiv.org/abs/1207.0580>.
- [11] L. Wan, M. Zeiler, S. Zhang et al., "Regularization of neural networks using Dropconnect," in *Proceedings of the International Conference on Machine Learning*, pp. 1058–1066, Atlanta, Georgia, June 2013.
- [12] A. Poernomo and D.-K. Kang, "Biased dropout and crossmap dropout: learning towards effective dropout regularization in convolutional neural network," *Neural Networks*, vol. 104, no. 1, pp. 60–67, 2018.
- [13] B. Cao, J. Li, and B. Zhang, "Regularizing neural networks with adaptive local drop," in *Proceedings of the 2015 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–5, IEEE, Killarney, Ireland, July 2015.
- [14] J. Tompson, R. Goroshin, A. Jain et al., "Efficient object localization using convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 648–656, Boston, MA, USA, June 2015.
- [15] I. J. Goodfellow, D. Warde-Farley, M. Mirza et al., "Maxout networks," 2013, <http://arxiv.org/abs/1302.4389>.
- [16] N. Srivastava, G. Hinton, A. Krizhevsky et al., "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.
- [17] S. Han, J. Pool, J. Tran et al., "Learning both weights and connections for efficient neural network," in *Proceedings of the Advances in Neural Information Processing Systems*, pp. 1135–1143, Montreal, Canada, December 2015.
- [18] A. Asuncion and D. Newman, *UCI Machine Learning Repository*, Springer, Berlin, Germany, 2007.
- [19] A. Iosifidis, A. Tefas, and I. Pitas, "DELM: DropELM: fast neural network regularization with dropout and DropConnect," *Neurocomputing*, vol. 162, no. 1, pp. 57–66, 2015.
- [20] R. Li, X. Wang, L. Lei et al., "L21-Norm based loss function and regularization extreme learning machine," *IEEE Access*, vol. 7, no. 1, pp. 6575–6586, 2018.
- [21] Y. Song, X. Wang, W. Quan et al., "A new approach to construct similarity measure for intuitionistic fuzzy sets," *Soft Computing*, vol. 23, no. 6, pp. 1985–1981, 2019.
- [22] B. Zhang, M. Zhang, Y. Song, and L. Zhang, "Combining evidence sources in time domain with decision maker's preference on time sequence," *IEEE Access*, vol. 7, no. 1, pp. 174210–174218, 2019.