RA-BLS: a Sequential BLSs Integrated with Residual Attention Mechanism

Yanqiang Wu, Jing Wang and Wei Hu

November 4, 2024

RA-BLS: a sequential BLSs integrated with residual attention mechanism

Yanqiang Wu[1], ✉Jing Wang[2], and Wei Hu[3]

School of Computer Science, Guangdong Polytechnic Normal University, Guangzhou 510665, China
Email: arolme@gpnu.edu.cn[1], wj_adr@163.com[2], awei@gpnu.edu.cn[3]

Abstract. Broad Learning System (BLS) has attracted the attention of many researchers because of its excellent performance. However, The number of random nodes becomes very large when the BLS copes with large complex datasets. Based on this problem, a novel residual attention mechanism is designed and introduced into the BLS to form a sequential BLSs integrated with residual attention mechanism (RA-BLS). The goal is to shrink the network structure by correcting random enhancement nodes. The RA-BLS feeds the residuals back to the enhancement nodes to get efficient and compact feature representations, further strengthening the approximation capability. The RA-BLS drastically reduces the node requirements and further improves the performance of BLS. Finally, experimental results show that RA-BLS achieves excellent performance on UCI, MNIST, FASHION-MNIST, and NORB datasets. Notably, an accuracy of 92.33% is achieved on the NORB dataset, marking an improvement of 2.99% with only 17% of the model size. The source code is available all at https://github.com/arolme/RABLS-Residual-Attention-Broad-Learning-System.

Keywords: Broad Learning System · Residual · Classification · Attention Mechanism · Machine Learning

1 Introduction

Broad Learning System [3], introduced by Chen, uses Ridge Regression to build a concise structure with an explicit solution. Because BLS can achieve [4] satisfactory performance quickly, it is widely applied [8] to address time-consuming tasks. The BLS employs a strategy of randomly generating hidden nodes, which necessitates a large number of these nodes [2, 5] to capture the information of the input units comprehensively. This approach can cause the BLS to become very fat [13, 15]. The fat BLS ensures thorough learning but impacts computational efficiency. Additionally, the fat BLS is also not conducive to practical application.

The core reason [1] for the fat BLS is that the random enhancement nodes are not designed based on the specific task. This means that different random nodes may capture similar or even duplicate information. Some scholars have proposed solutions to the problem of conflict between model size and performance. The practice is to keep the compact network. Ding [6] proposed a greedy BLS that reduces the redundant nodes in the hidden layer, constructing a structure that compromises width and depth. In D&BLS [19], Xie passes the residuals to the next subsystem. Furthermore, multiple lightweight BLSs are stacked to replace the fat BLS.

Our motivation is to address the fat BLS via a compact structure. Inspired by the D&BLS, we sequence multiple lightweight BLSs to replace the fat BLS. We propose a sequential Broad Learning System with residual attention mechanism (RA-BLS) to address fat BLS caused by excessive random nodes. Unlike D&BLS, the lightweight subsystem in RA-BLS does not require the regeneration of hidden nodes. Instead, the hidden nodes of RA-BLS are transferred from lower to upper layers, facilitating their reuse and potentially improving computational efficiency. In addition, inspired by attention mechanism [17], we feed the residuals back into the enhancement nodes of the subsystem, thereby weakening the randomness of the enhancement nodes. The aim is to adapt the enhancement nodes to a specific task. The hidden nodes of the uppermost layer are directly connected to the output, with the output weights calculated by the pseudoinverse. The key contributions are outlined:

1. The new system introduces a residual attention mechanism to correct the enhancement nodes according to the current task. This method leads to the creation of slim and compact subsystems, which improves the system's efficiency.

2. A small-scale network is maintained by forming a sequential model through multiple compact subsystems. The optimal output of the last subsystem is generated after passing through multiple subsystems. This innovative approach corrects the enhancement nodes to adapt to the current task rather than capture random features. The fatness problem of BLS is solved, further improving the superior accuracy on large and complex datasets.

The structure is as follows as follows: Section 2 briefly describes the background of BLS and the attention mechanism. The residual attention mechanism and the structure of RA-BLS are detailed in Sections 3. Experimental results are shown in Section 4. The performance and the generalization capability of RA-BLS are compared with other popular methods. The effects of different parameters are also discussed. Finally, Section 5 offers conclusions and suggests directions for future research.

2 Related Works

2.1 Brief Introduction of Broad Learning System

The BLS [3] contains (see Fig. 1) three main structures: input unit, hidden layer, and output layer. Two distinct node types are present within the hidden layer: feature nodes and enhancement nodes. The core idea involves a mapping function that maps raw input units into feature nodes. These feature nodes are then augmented to generate enhancement nodes randomly. Finally, each hidden node is directly connected to the output, with the pseudoinverse calculated output weights.

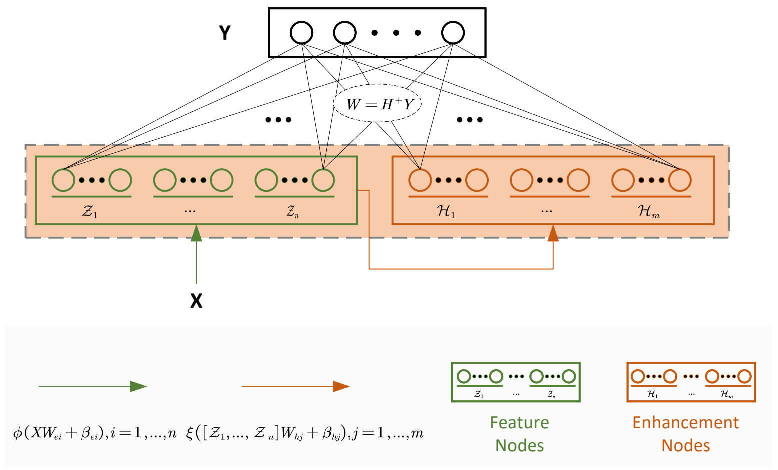

Fig. 1. Illustration of the BLS.

For the original BLS, a given input $\{(X,Y)|X \in \mathbb{R}^{N\times M}, Y \in \mathbb{R}^{N\times C}\}$. Let $\phi(\cdot)$ denote the mapping function for n feature nodes, and $\xi(\cdot)$ denote the activation function that produces m enhancement nodes. The mapping function $\phi(\cdot)$ and activation function $\xi(\cdot)$ have no explicit restrictions, which means that the common choices such as kernel mappings or nonlinear transformations are acceptable. Specifically, if the function in the feature mapping uses kernel mappings, the BLS has additional connecting nodes called enhancement nodes between the feature mapping layers and the output layer. The feature information hidden in the enhancement nodes can be further explored. The n feature nodes and m enhancement nodes can be collectively represented as hidden nodes:

$$H = [\mathcal{Z}_1, \ldots, \mathcal{Z}_n, \mathcal{H}_1, \ldots, \mathcal{H}_m]. \tag{1}$$

where, using $Z_i = \phi(XW_{ei} + \beta_{ei}), i = 1, \ldots, n$ to denote the ith Z_i mapped features, and denoting all feature nodes as $[Z_1, \ldots, Z_n]$. And denoting $H_j = \xi([Z_1, \ldots, Z_n]W_{hj} + \beta_{hj}), j = 1, \ldots, m$ as the jth enhancement nodes, where $W_{ei}, \beta_{ei}, W_{hj}$ and β_{hj} are generated randomly with corresponding dimensions.

The output of BLS can be represented as the equation of the form

$$Y = [\mathcal{Z}_1, \dots, \mathcal{Z}_n, \mathcal{H}_1, \dots, \mathcal{H}_m]W. \tag{2}$$

W represents the connecting weights between the hidden nodes and the output. The pseudoinverse is expressed as $H^+ = (\lambda I + H^T H)^{-1} H^T$. The regression approximation can be easily computed as W using the equation

$$W = (\lambda I + H^T H)^{-1} H^T Y. \tag{3}$$

2.2 Attention Mechanism

Attention [17] is inspired by the biological system. Humans actively focus their attention on a certain part when processing a large amount of information. Attention mechanisms are widely applied in deep learning.

In deep learning, the attention mechanism is designed according to the specific tasks. The **query** and **key** are designed according to the particular task or network structure. The correlation between the **key** and the **query** reflects the importance of the attention. A matrix Q is packed with all **queries**. Similarly, the **keys** and **values** are also packed together into matrices K and V. The attention mechanism is computed in the matrix of outputs as:

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{D}})V. \tag{4}$$

where Q and K are computed dot product, divide each by $\sqrt{D}$, and the softmax function is applied to the score of V. D denotes the dimension of the feature. The $\frac{1}{\sqrt{D}}$ is used as a scaling factor to counteract the effect of large scores.

3 The Proposed Methods

3.1 Residual Attention Mechanism

The BLS is a discriminative model. The decision boundary of BLS is searched to separate different categories. The residual [12] represents the distance between the predicted value and the true value. The smaller the residual is, the clearer the decision boundary is. Therefore, we propose residual attention mechanisms to correct the random enhancement nodes to fit a given task.

$$\text{Attention}(Q, K, R) = \text{softmax}(\frac{QK^T}{\sqrt{D}})R. \tag{5}$$

where R represents the residual of the current state, and D represents the dimension of the feature.

3.2 BLS with Residual Attention Mechanism

In RA-BLS, the hidden nodes of the lowest layer are randomly generated. The hidden nodes are passed layer by layer for reuse. Two distinct node types are present within the hidden nodes: feature nodes and enhancement nodes. The feature nodes are retained in the next subsystem. The enhancement nodes have their randomness weakened by the residual attention mechanism. Immediately after that, the feature nodes and the corrected enhancement nodes are combined to form the new hidden nodes. These hidden nodes are connected to the output.

For the RA-BLS, mathematically, we denote the input $\{(X, Y) | X \in \mathbb{R}^{N \times M}, Y \in \mathbb{R}^{N \times C}\}$. First, recall that the original BLS can be represented equivalently by Eq. (1)-(3). Note that the nodes for the first iteration are generated randomly. The feature nodes $F = [\mathcal{Z}_1, \dots, \mathcal{Z}_n]$, inactive enhancement nodes $E^{(1)}_{inactive} = [(FW_{h1} + \beta_{h1}), \dots, (FW_{hm} + \beta_{hm})]$, enhancement nodes $E^{(1)}_{active} = \xi(E^{(1)}_{inactive})$, weights $W^{(1)}$ and residual $R^{(1)}$ for the first iteration are easily obtained.

Then, the process of optimizing the enhancement nodes begins. The weights are split into two parts

$$\begin{bmatrix} W_f^{(1)} \\ W_e^{(1)} \end{bmatrix} = W^{(1)}. \tag{6}$$

where $W_e^{(1)}$ is a weight that connects the enhancement nodes to the output. Similarly, $W_f^{(1)}$ is a weight that connects the feature nodes to the output. So that the output is equivalent to $Output^1 = [F, E_{active}^{(1)}] \begin{bmatrix} W_f^{(1)} \\ W_e^{(1)} \end{bmatrix} = FW_f^{(1)} + E_{active}^{(1)} W_e^{(1)}$.

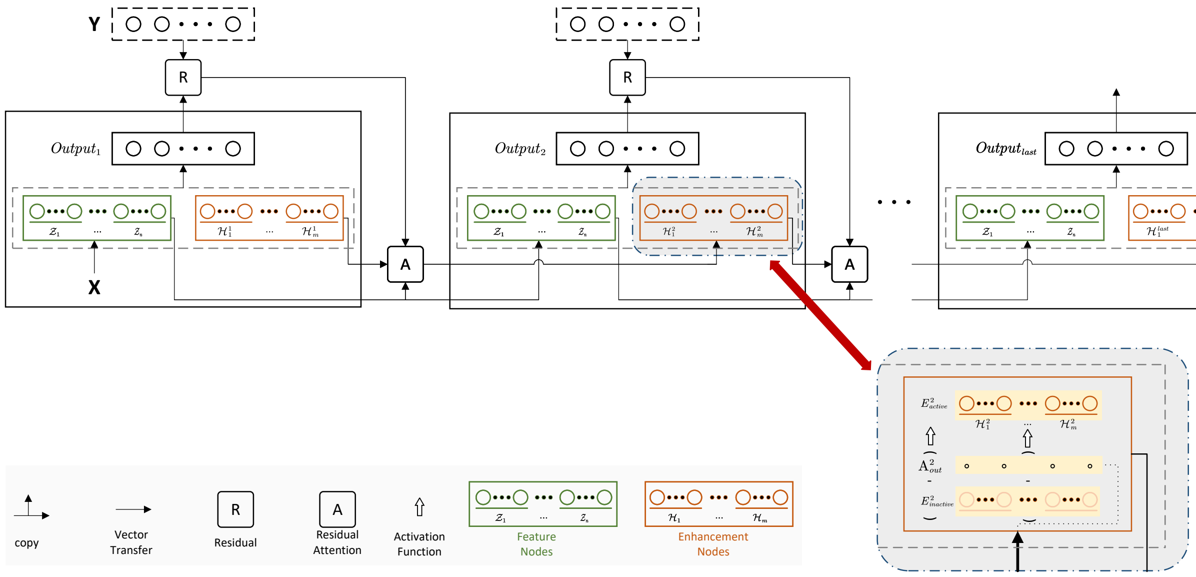
Y
R
Output$_1$
$\mathcal{Z}_1$ ... $\mathcal{Z}_s$
$\mathcal{H}_1^1$ ... $\mathcal{H}_m^1$
X
A
Output$_2$
$\mathcal{H}_1^2$ ... $\mathcal{H}_m^2$
Output$_{last}$
$\mathcal{H}_1^{last}$ ... $\mathcal{H}_m^{last}$
E_{active}^2
A_{out}^2
$E_{inactive}^2$
copy
Vector Transfer
Residual
Residual Attention
Activation Function
Feature Nodes
Enhancement Nodes
$\mathcal{H}_1$ ... $\mathcal{H}_m$

Fig. 2. Lightweight subsystems are sequenced. All nodes are randomly generated in the first subsystem. Each subsystem reuses feature nodes. Only the enhancement nodes of the lower layer have their randomness weakened by the residual attention mechanism. Immediately after, the nodes are fed to the upper layer subsystem.

In RA-BLS (See Fig. 2), the feature nodes F represent the matrix K and matrix Q. $\mathrm{A}(Q, K, R, W) = (\bigtriangledown\xi) \circ \mathrm{softmax}(\frac{FF^T}{\sqrt{D}})RW^T$ is the residual attention block. $E_{inactive}$ denotes an inactive enhancement node. The core of correction is to fuse the residual information before the enhancement node is activated. The equation for (t+1)th correction-based residual attention is expressed as:

$$E_{inactive}^{(t+1)} = E_{inactive}^{(t)} - (\bigtriangledown\xi) \circ \mathrm{softmax}(\frac{FF^T}{\sqrt{D}})R^{(t)} W_e^{(t)T}. \tag{7}$$

where $R^{(t)}$ denotes the residuals of (t)th iteration. $\circ$ denotes the Hadamard product. $(\bigtriangledown\xi)$ denotes the gradient of the activation function $\xi(\cdot)$. D denotes the dimension of the feature nodes. The purpose of introducing the $(\bigtriangledown\xi)$ is to counteract the effect of the activation function in the residual pass. The W_e^t is a weight that connects the enhancement nodes to the output. The role of W_e^T is to communicate the residual back to the corresponding enhancement nodes.

Next, the corrected nodes are activated to form enhancement nodes

$$E_{active}^{(t+1)} = [\mathcal{H}_1^{(t+1)}, \ldots, \mathcal{H}_m^{(t+1)}] = \xi(E_{inactive}^{(t+1)}). \tag{8}$$

The $H^{(t+1)} = [\mathcal{Z}_1, \ldots, \mathcal{Z}_n, \mathcal{H}_1^{(t+1)}, \ldots, \mathcal{H}_m^{(t+1)}]$ denotes as the hidden nodes of the (t+1)th iteration. And the weights $W^{(t+1)} = (\lambda I + H^{(t+1)T} H^{(t+1)})^{-1} H^{(t+1)T} Y$ are calculated using pseudoinverse. Then, the output is obtained

$$Output^{last} = H^{(last)} W^{(last)}. \tag{9}$$

The RA-BLS is an optimization algorithm based on the residual attention mechanism. The number of subsystems is also a very important hyperparameter. In general, too much attention to residuals can cause overfitting. Therefore, we set the iteration to terminate when the performance drops off.

Algorithm 1 RA-BLS

```
Require: The samples $\{(X,Y) | X \in \mathbb{R}^{N \times M}, Y \in \mathbb{R}^{N \times C}\}$, mapping function $\phi(\cdot)$, activation function $\xi(\cdot)$, number of Feature
    Nodes $n$, number of Enhancement Nodes $m$ and regularization coefficient $\lambda$.
Ensure: $Output_{last}$
 1: Initialize parameters;
 2: Randomly generate $n$ Feature Nodes $\mathcal{Z}_i \Leftarrow \phi(XW_{ei} + \beta_{ei}), i = 1, \ldots, n$;
    % Matrix of all Feature Nodes $F = [\mathcal{Z}_1, \ldots, \mathcal{Z}_n]$.
 3: Randomly generate $m$ Enhancement Nodes $\mathcal{H}_j \Leftarrow \xi(FW_{hj} + \beta_{hj}), j = 1, \ldots, m$; % Feature Nodes are mapped to
    get inactive Enhancement Nodes $E^{(1)}_{inactive} = [(FW_{h1} + \beta_{h1}), \ldots, (FW_{hm} + \beta_{hm})]$. Immediately after that, a nonlinear
    activation function is used to activate Enhancement Nodes $E^{(1)}_{active} = \xi(E^{(1)}_{inactive})$.
 4: Combine the Hidden Nodes $H^{(1)} \Leftarrow [F, E^{(1)}_{active}]$;
 5: Calculate the Weight $W^{(1)} \Leftarrow (\lambda I + H^{(1)T} H^{1})^{-1} H^{(1)T} Y$;
 6: Calculate the output $Output^{(1)} \Leftarrow H^{(1)} W^{(1)}$;
 7: Calculate the Residual $R^{(1)} \Leftarrow (Y - Output^{(1)})$;
 8: $t \Leftarrow 1$;
 9: while $True$ do
10:     Split the weight corresponding to Enhancement Nodes $W_e^{(t)} \Leftarrow W^{(t)}$ by Eq. 6;
11:     Calculate new Enhancement Nodes $E^{(t+1)}_{active} \Leftarrow \xi(E^{(t)}_{inactive} - (\bigtriangledown \xi) \circ \mathrm{softmax}(\frac{FF^T}{\sqrt{D}}) R^{(t)} W_e^{(t)T})$;%D denotes the dimen-
    sion of the feature nodes.
12:     Combine new Hidden Nodes $H^{(t+1)} = [F, E^{(t+1)}_{active}]$
13:     Calculate new Weight $W^{(t+1)} \Leftarrow (\lambda I + H^{(t+1)T} H^{t+1})^{-1} H^{(t+1)T} Y$;
14:     Calculate new output $Output^{(t+1)} \Leftarrow H^{(t+1)} W^{(t+1)}$;
15:     Calculate new Residual $R^{(t+1)} \Leftarrow (Y - Output^{(t+1)})$;
16:     $t \Leftarrow t + 1$;
17:     if performance drops then
18:         Return $Output_{last} \Leftarrow Output^{(t)}$
19:     end if
20: end while
```

4 Experiments

This section details the experiments of the RA-BLS. Various methods are also compared on several popular datasets.

In our paper, function $\phi(\cdot)$ uses a linear feature mapping in the proposed RA-BLS. For the enhancement nodes, the Tanh function $\xi(\cdot)$ is chosen to build the RA-BLS. The regularization coefficient λ for RA-BLS, BLS, Stacked BLS, and ER-BLS is set as 2^{-30} by grid search. The W_{hj} and β_{hj}, for $j = 1, \ldots, m$ obey the standard uniform distribution on the interval [-1,1]. In addition, we set the iteration to terminate when the performance drops off.

4.1 Experiment Environment and Assessment Metric

The experiments for RA-BLS are given. To make it more convincing, the results are compared on several authoritative public datasets using different methods. The accuracy is averaged over 20 experiments. Our source code will provide frozen random seeds to fix the experimental results. The fixed results are slightly higher or equal to the paper results.

The RA-BLS is evaluated against several methods, including SVM with RBF Kernel [9], BLS [3], Stacked BLS [14], and ER-BLS [7]. The datasets include MNIST [10], FASHION-MNIST [18], NORB [11], and UCI datasets. The RA-BLS is compared with the current popular methods on these datasets. The SVM does not use special feature selection. Regarding experimental settings, all experiments in this paper are implemented using MATLAB 2021a.

Accuracy is the most straightforward and widely adopted metric for evaluating performance in classification tasks. The Accuracy can be represented as the equation of the form:

$$Accuracy = \frac{\text{total correct predictions}}{\text{total samples}} \times 100\%. \tag{10}$$

For the regression task, the evaluating metric is Root Mean Squared Error. The equation is:

$$RMSE = \sqrt{\frac{1}{k}\sum_{t=1}^{k}(\mathcal{Y}_i - \hat{\mathcal{Y}}_i)^2}. \tag{11}$$

.

4.2 Performance Evaluation on Classification

BLS is experimentally demonstrated on MNIST. In addition to MNIST, our experiments utilize the FASHION-MNIST dataset, which presents a more challenging classification task. The FASHION-MNIST dataset maintains the same structure as MNIST but with increased complexity. The NORB dataset is primarily utilized to assess the efficacy of computer vision algorithms in 3D object recognition tasks. Table 1 presents details regarding the datasets used.

Table 1. Classification dataset details

	No. of Samples			
Datasets	Training	Testing	Features	No. of Categories
MNIST	60000	10000	$768(28 \times 28)$	10
FASHION-MNIST	60000	10000	$768(28 \times 28)$	10
NORB	24300	24300	$2048(32 \times 32 \times 2)$	5

The results are presented in Table 2. The number of feature nodes N_f and enhancement nodes N_e are determined through an exhaustive search from the ranges of [1, 2000] and [100, 15000]. To ensure fairness in the results. The feature engineering of the RA-BLS remains the same as that of the BLS and only updates the enhancement node proposed in our method. At the same time, the N_f and the hyperparameter settings are kept the same.

Table 2. Comparative Analysis of SVM, BLS, Stacked BLS, ER-BLS and RA-BLS for Classification

	MNIST			FASHION-MNIST			NORB		
Method	N_f	N_e	Acc(%)	N_f	N_e	Acc(%)	N_f	N_e	Acc(%)
SVM	-	-	97.92	-	-	88.28	-	-	84.49
BLS(2017)	1000	15000	98.72	1500	9000	89.73	1500	9000	89.34
Stacked BLS(2021)	1000	9000	98.91	1000	15000	89.59	1000	3000	89.67
ER-BLS(2023)	100	9000	98.94	200	8000	90.29	1600	2000	89.18
RA-BLS(ours)	1000	2000	**99.02**	1500	2000	**91.07**	1500	300	**92.33**

Table 2 shows that RA-BLS consistently achieves the best accuracy. The Accuracy on MNIST, FASHION-MNIST and NORB datasets are **99.02%**, **91.07%** and **92.33%**. This is because the performance of RA-BLS improves as the randomness of the enhancement nodes is iteratively weakened. Remarkably, the RA-BLS even improves accuracy, while the structure size is only 17% of that of BLS on the NORB dataset. This result indicates that RA-BLS can significantly reduce the model size while maintaining high performance. The RA-BLS demonstrates a significant advantage when higher accuracy is sought in the NORB dataset. The results confirm that RA-BLS outperforms BLS and prove that it is promising and effective.

4.3 Performance Evaluation on Regression

To evaluate the performance of the RA-BLS, three regression datasets from the UCI are selected for analysis. Table 3 presents details regarding the datasets used.

Table 3. Regression dataset details

	No. of Samples		
Datasets	Training	Testing	Features
Basketball	62	32	8
Mortgage	699	350	15
Housing	337	169	13

Table 4. Performance comparison of SVM, BLS and RA-BLS for Regression

	Basketball			Mortgage			Housing		
Method	N_f	N_e	RMSE	N_f	N_e	RMSE	N_f	N_e	RMSE
SVM	-	-	0.0894	-	-	0.0485	-	-	0.0989
BLS(2017)	21	1	0.0791	50	130	0.0051	145	50	0.0748
RA-BLS(ours)	21	1	**0.0790**	50	130	**0.0047**	145	50	**0.0742**

BLS and RA-BLS maintain the same number of nodes for low-complexity regression tasks. The parameters N_f and N_e have specified search ranges of [1,150] and [1,200]. The RA-BLS performs excellently (See Table 4) on regression datasets and BLS. The proposed RA-BLS demonstrates superior performance on the Mortgage dataset. The results demonstrate the effectiveness of the RA-BLS.

5 Conclusion

This paper proposes a compact network called RA-BLS and performs an exhaustive experimental evaluation. The new system adopts a residual attention mechanism to correct the enhancement nodes to adapt to the current task rather than capture random features. The enhancement nodes without randomness significantly improve the efficiency and performance of BLS. This design replaces the fat BLS with a more compact and efficient structure. In addition, the RA-BLS provides a highly efficient and robust framework for dealing with various complex data tasks. The method for the residual attention mechanism is interchangeable, such as methods related to other attention mechanisms or reducing redundancy. The RA-BLS demonstrates the potential of BLS in modern data science and opens up new possibilities for future research and applications.

However, RA-BLS uses linear mapping to generate feature nodes. It cannot handle ultra-complex datasets such as CIFAR100 or ImageNet well. For ultra-complex datasets, the feature nodes should be generated by a more advanced deep learning feature extractor. Exploring the combination of advanced deep learning and RA-BLS is also a topic of great significance.

Funding The work was supported by the Research Platforms and Program of the Education Department of Guangdong Province in China [grant numbers 2023ZDZX1011 and 2022ZDZX1013], and Scientic and Technological Planning Projects of Guangdong Province [grant numbers 2021A0505030074]. It was also supported partly by Guangdong Basic and Applied Basic Research Foundation [grant numbers 2021A1515011999], and the Guangdong Provincial Key Laboratory of Traditional Chinese Medicine Informatization [grant numbers 2021B1212040007].

References

1. Asi, H., Duchi, J.C.: The importance of better models in stochastic optimization. Proceedings of the National Academy of Sciences **116**(46), 22924–22930 (2019)
2. Chen, C.L.P.: A rapid supervised learning neural network for function interpolation and approximation. IEEE Transactions on Neural Networks **7**(5), 1220–1230 (1996)
3. Chen, C.L.P., Liu, Z.: Broad learning system: An effective and efficient incremental learning system without the need for deep architecture. IEEE transactions on neural networks and learning systems **29**(1), 10–24 (2017)
4. Chen, C.L.P., Liu, Z., Feng, S.: Universal approximation capability of broad learning system and its structural variations. IEEE transactions on neural networks and learning systems **30**(4), 1191–1204 (2018)
5. Chen, C.L.P., Wan, J.Z.: A rapid learning and dynamic stepwise updating algorithm for flat neural networks and the application to time-series prediction. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) **29**(1), 62–72 (1999)
6. Ding, W., Tian, Y., Han, S., Yuan, H.: Greedy broad learning system. IEEE Access **9**, 79307–79315 (2021)
7. Gan, J., Xie, X., Zhai, Y., He, G., Mai, C., Luo, H.: Facial beauty prediction fusing transfer learning and broad learning system. Soft Computing **27**(18), 13391–13404 (2023)
8. Gong, X., Zhang, T., Chen, C.L.P., Liu, Z.: Research review for broad learning system: Algorithms, theory, and applications. IEEE Transactions on Cybernetics **52**(9), 8922–8950 (2021)
9. Han, S., Qubo, C., Meng, H.: Parameter selection in svm with rbf kernel function. In: World Automation Congress 2012. pp. 1–4 (2012)
10. Le Cun, Y., Boser, B., Denker, J.S., Henderson, D., Howard, R.E., Hubbard, W., Jackel, L.D.: Handwritten digit recognition with a back-propagation network. In: Proceedings of the 2nd International Conference on Neural Information Processing Systems. vol. 9, p. 396–404. MIT Press, Cambridge, MA, USA (1989)
11. LeCun, Y., Huang, F.J., Bottou, L.: Learning methods for generic object recognition with invariance to pose and lighting. In: Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004. vol. 2, pp. 2–104 (2004)
12. Ling, R.F.: Residuals and influence in regression. Technometrics **26**(4), 413–415 (1984)
13. Liu, Z., Chen, B., Xie, B., Qiang, H., Zhu, Z.: Feature selection for orthogonal broad learning system based on mutual information. In: 2019 International Joint Conference on Neural Networks (IJCNN). pp. 1–8. IEEE (2019)
14. Liu, Z., Chen, C.L.P., Feng, S., Feng, Q., Zhang, T.: Stacked broad learning system: From incremental flatted structure to deep model. IEEE Transactions on Systems, Man, and Cybernetics: Systems **51**(1), 209–222 (2020)
15. Ma, J., Fan, J., Wang, L., Chen, C.P., Yang, B., Sun, F., Zhou, J., Zhang, X., Gao, F., Zhang, N.: Factorization of broad expansion for broad learning system. Information Sciences **630**, 271–285 (2023)
16. Pao, Y.H., Takefuji, Y.: Functional-link net computing: theory, system architecture, and functionalities. Computer **25**(5), 76–79 (1992)
17. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need (2023), https://arxiv.org/abs/1706.03762
18. Xiao, H., Rasul, K., Vollgraf, R.: Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms (2017), https://arxiv.org/abs/1708.07747
19. Xie, R., Wang, S.: Downsizing and enhancing broad learning systems by feature augmentation and residuals boosting. Complex & Intelligent Systems **6**, 411–429 (2020)