ISSN 2616-5775 Vol. 7, Issue 2: 48-58, DOI: 10.25236/AJCIS.2024.070207

Recognition of Steel Surface Defects Based on Broad Learning System

Zhonghao Qiu[1], Xixin Yang[1,2,a,*], Yuanlin Guan[3,4,b,*], Xiang Yuan[1]

[1]*College of Computer Science & Technology, Qingdao University, Qingdao, China*
[2]*School of Automation, Qingdao University, Qingdao, China*
[3]*Key Lab of Industrial Fluid Energy Conservation and Pollution Control, Ministry of Education, Qingdao University of Technology, Qingdao, China*
[4]*School of Mechanical & Automotive Engineering, Qingdao University of Technology, Qingdao, China*
[a]*yangxixin@qdu.edu.cn,* [b]*guanyuanlin@qut.edu.cn*
[*]*Corresponding author*

Abstract: *With the development of Industry 4.0, real-time industrial inspection has been widely focused. Steel is a critical material, and steel surface defect recognition is of great significance to the production in plants. However, defect recognition models are usually based on deep learning methods, which leads to long training time and high hardware requirements. Hence, this article proposes a novel recognition scheme for steel surface defects based on Broad Learning System (BLS). This model takes the image features extracted by the convolutional neural network as input data and inherits the advantages of the BLS, which can classify images quickly. The results indicate that the new method outperforms both the original BLS and several mainstream algorithms.*

Keywords: *Steel Surface, Defect Recognition, Broad Learning System, Convolutional Neural Network, Incremental Learning, Dropout Method*

1. Introduction

In numerous industries including manufacturing, energy, and transportation, steel plays a critical role as a common material. Unfortunately, factors such as equipment fatigue and external forces can lead to the emergence of diverse defects on the surface of steel, such as patches, inclusions and scratches [1]. These defects directly impact the load-bearing capacity and service life of steel products and can even initiate chain reactions, posing significant safety hazards [2]. Hence, the timely inspection of these defects holds immense importance.

Surface defect recognition in the context of steel is predominantly carried out through manual classification [3, 4]. However, to enhance efficiency and mitigate production costs, there is a growing impetus for automating the recognition of steel surface defects. Consequently, the proficient implementation of artificial intelligence for accurately classifying surface defects in steel has emerged as a prevailing necessity in industrial plants [5].

Konovalenko et al. [6] utilized the deep residual neural network to construct a model for high-precision surface defect recognition. The hybrid network architecture (CNN-T) [7] integrated Convolutional Neural Network (CNN) and Transformer encoder, exhibiting strong inductive biases and global modeling capability. In scenarios where surface defect samples are scarce, a dual-stream neural network [8] introduced sample generation and transfer learning methods, exploring the use of CNN to enhance the quality of generated defect images. The Concurrent Convolutional Neural Network (ConCNN) [9] tackled this issue by leveraging various image scales and a fusion strategy, and can learn new incoming defect types online. Some studies have focused on feature representation. Fu et al. [10] introduced a compact yet effective CNN model that emphasizes learning low-level features of defect images and tested it on a diversity-enhanced dataset. The Feature-aware Network (FaNet) [11] adopted a feature-attention convolution module to extract the comprehensive feature information from the base classes.

Nonetheless, a significant drawback of these existing approaches is their heavy reliance on substantial amounts of time and resources for model training. As a consequence, the application of these approaches becomes challenging in small and medium-sized plants that possess limited hardware capabilities.

Residual Network (ResNet) [12] is a deep convolutional neural network architecture proposed by

ISSN 2616-5775 Vol. 7, Issue 2: 48-58, DOI: 10.25236/AJCIS.2024.070207

Kaiming He et al., and it is also an important milestone in the field of computer vision. ResNet enables the seamless propagation of information between network layers, thereby enhancing model accuracy and generalization capability. However, it still inherits the common challenge of lengthy training time associated with deep neural networks.

Broad Learning System (BLS) [13] is a novel machine learning model introduced by Prof. C. L. Philip CHEN in 2018, based on the Random Vector Functional Link Neural Network (RVFLNN). Unlike conventional deep learning approaches, BLS addresses the challenges of lengthy training time and high hardware requirements by employing a broad structure and fast computation methods. However, it has been observed that training and testing BLS solely using raw data may not fully harness its performance potential.

Building upon previous research, this paper presents an innovative machine learning methodology for the inspection of steel surface defects, leveraging a fusion of the Broad Learning System and the convolutional neural network ResNet-18. This approach facilitates swift and precise recognition, with the capacity for ongoing enhancement.

2. Related Work

2.1. ResNet-18 Network

The ResNet series networks are widely used in computer vision tasks because of their superior performance. Due to the introduction of residual blocks, the degradation caused by the increase in the number of layers is well resolved by ResNet.

The key distinguishing factor among ResNet networks, which include ResNet-18, ResNet-34 and ResNet-50, is the number of layers within each network. ResNet-18 is comparatively shallow, as shown in Figure 1. It demonstrates similar performance to other ResNet networks, while having the capability to preserve a greater amount of low-scale features. Considering the hardware limitations of small and medium-sized plants, as well as the practical performance of the model, ResNet-18 has been chosen for application in this paper. The selection rationale will be further discussed in the experimental and result analysis section.

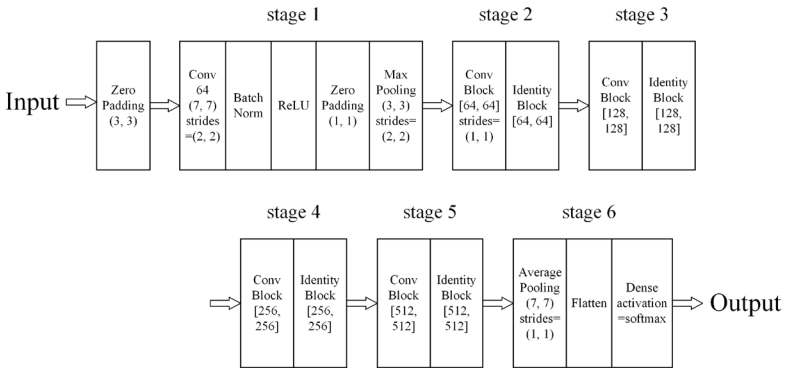

Figure 1: Residual Network (ResNet-18) structure.

2.2. Basic Model of Broad Learning System

Compared to deep networks, the BLS exhibits a simpler structure, primarily consisting of mapped feature nodes, enhancement nodes, and output weights.

The input data is denoted as X, which passes through the model and generates multiple groups of mapped feature nodes through mapping.

$$Z_i = \phi\left(XW_{e_i} + \beta_{e_i}\right), \quad i = 1, \dots, n \tag{1}$$

where Z_i denotes the ith group of mapped feature nodes, W_{e_i} and β_{e_i} are the random weights and biases with the proper dimensions, and ϕ indicates linear transformation. $Z^n = [Z_1, \dots, Z_n]$ is the concatenation of n groups of mapped feature nodes, on which the enhancement nodes are generated.

$$E_j = \xi\left(Z^n W_{h_j} + \beta_{h_j}\right), \quad j = 1, \dots, m \tag{2}$$

ISSN 2616-5775 Vol. 7, Issue 2: 48-58, DOI: 10.25236/AJCIS.2024.070207

where E_j represents the jth group of enhancement nodes, W_{h_j} and β_{h_j} denote the random weights and biases respectively, with the appropriate dimensions. The variable ξ indicates linear transformation. Thus, the concatenation of m groups of enhancement nodes is denoted as $E^m = [E_1, \dots, E_m]$.

The number of nodes per group v and the number of groups n for mapped feature nodes, as well as the number of nodes per group η and the number of groups m for enhancement nodes in the BLS, are selected based on the complexity of the specific task.

$$\begin{aligned} Y &= \left[Z_1, \dots, Z_n | \xi\left(Z^n W_{h_1} + \beta_{h_1}\right), \dots, \xi\left(Z^n W_{h_m} + \beta_{h_m}\right)\right] W \\ &= [Z_1, \dots, Z_n | E_1, \dots, E_m] W \\ &= [Z^n | E^m] W \\ &= HW \end{aligned} \tag{3}$$

Formula (3) represents the basic model of BLS, where H is the concatenation of all mapped feature nodes and enhancement nodes, while Y denotes the data labels. W corresponds to the output weights, which serves as the training objective as well. A graphical depiction of this model is illustrated in Figure 2.

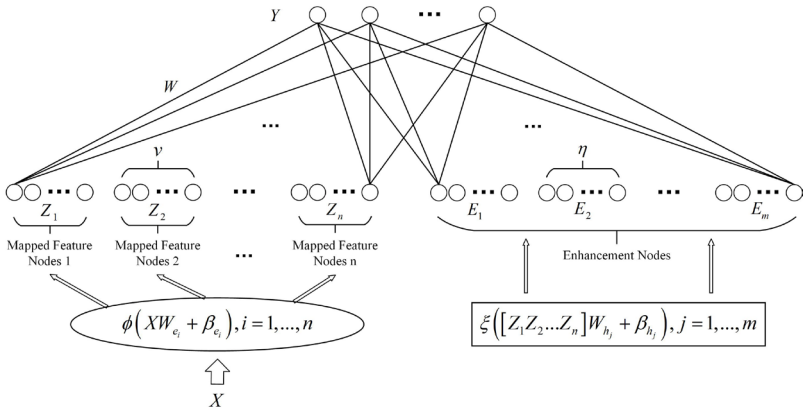

Figure 2: Basic model of Broad Learning System (BLS).

To solve for the output weights, W, that minimize the training error, ridge regression can be employed, as shown in Formula (4).

$$W = \begin{cases} H^{\mathrm{T}}(cI + HH^{\mathrm{T}})^{-1}Y, & K < L \\ (cI + H^{\mathrm{T}}H)^{-1}H^{\mathrm{T}}Y, & K \geq L \end{cases} \tag{4}$$

where $L = nv + m\eta$ represents the total number of mapped feature nodes and enhancement nodes, while K denotes the number of input samples. Parameters c and I correspond to the regularization factor and L-order unit matrix, respectively.

3. Proposed Models

3.1. Convolutional Broad Learning System

In response to the problem that existing steel surface defect recognition models are inefficient to train and rely on high-specification hardware to the extent that they are difficult to deploy in most plants, we propose a fusion model called Convolutional Broad Learning System (C-BLS). The specific structure of the proposed model is illustrated in Figure 3.

ISSN 2616-5775 Vol. 7, Issue 2: 48-58, DOI: 10.25236/AJCIS.2024.070207

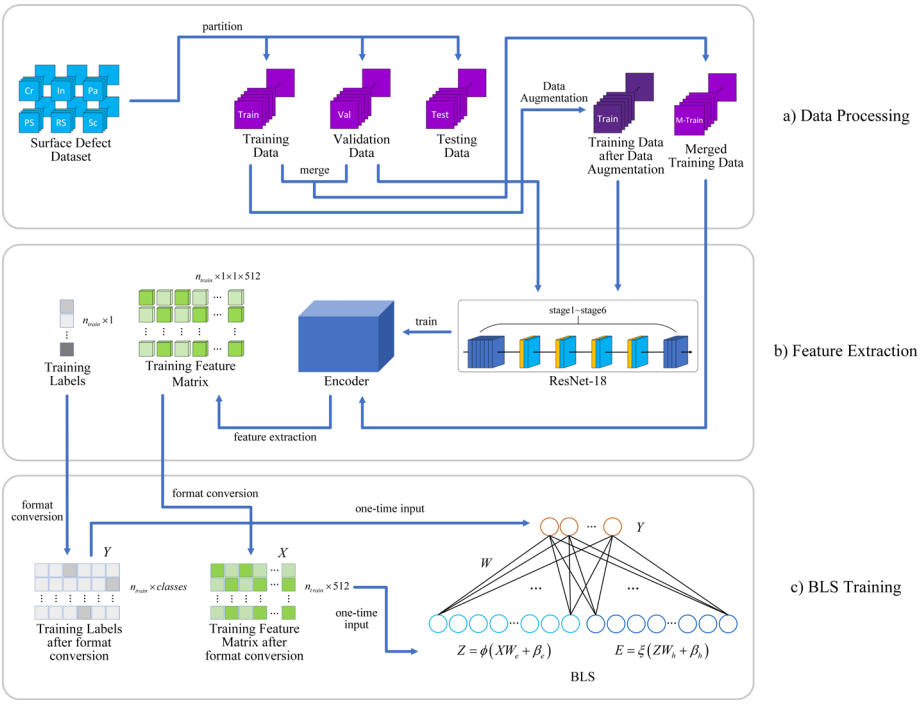

Figure 3: Convolutional Broad Learning System (C-BLS) structure.

Firstly, the surface defect dataset is partitioned into three distinct sets: training data, validation data, and testing data. Data augmentation techniques are applied to the training data to enrich its diversity and enhance its utility. Concurrently, the original training data and validation data are merged to create a unified dataset for training the BLS.

Next, the data augmented training data and validation data are input into the ResNet-18 for initial training. Unlike traditional CNN models that require numerous epochs to achieve high accuracies, this process is intentionally kept brief. The objective here is to equip ResNet-18 with basic feature extraction capabilities rather than focusing extensively on this step, as it is not directly utilized for the classification task.

Following the initial training, stage1 to stage5, along with the global average pooling layer in ResNet-18, are encapsulated as an encoder. This encoder is employed to extract features from the image data. The global average pooling layer compresses the feature maps of each channel into a single value while preserving important feature information, significantly reducing the data volume.

Subsequently, the merged training data is fed into the encoder to extract image features and generate a training feature matrix. To ensure smooth input into the BLS, the matrix undergoes format conversion by removing two redundant dimensions. Simultaneously, the labels are transformed into a one-hot encoded matrix. Finally, both matrices, denoted as X and Y, are input into the BLS for training at once. This mechanism is fast and efficient. Once training concludes, the random weights, random biases, and connection weights are fixed and will be used for the testing phase. The specific testing method will be introduced in the next section.

C-BLS is the fundamental model proposed by us for surface defect recognition. In C-BLS, the image data is not directly incorporated into the BLS. Instead, it undergoes a feature extraction process through a convolutional structure, resulting in a feature matrix that contains important location information. This approach addresses the issue of inadequate learning that arises when image data is directly fed into the BLS for image classification tasks, as observed in previous studies. Leveraging the robust learning capabilities of the BLS, the recognition performance of C-BLS surpasses that of ResNet-18 after initial training. Notably, the C-BLS training is efficient and does not rely on advanced hardware support. By resorting to the solution of the ridge regression problem, it can be performed on most standard computer systems.

3.2. Convolutional Broad Learning System with Dropout Incremental Learning

In the testing process of C-BLS, the encoder performs feature extraction on the testing data,

ISSN 2616-5775 Vol. 7, Issue 2: 48-58, DOI: 10.25236/AJCIS.2024.070207

generating a testing feature matrix. After undergoing format conversion, the matrix is input into BLS in a single iteration, and the final recognition result is obtained by computing it with the random weights, random biases, and connection weights determined during the training process.

However, when the accuracy of the recognition result does not meet the desired criteria, further reinforcement of the model is necessary to enhance its performance. Incremental learning is a commonly employed approach in this regard. In the case of BLS, incremental learning is achieved by adding nodes. The unique structure of the BLS makes it particularly suitable for incremental learning, enabling the rapid computation of connection weights for the new nodes. The addition of enhancement nodes represents the most used approach for incremental learning within the BLS [14].

The integration of incremental learning introduces complexity to the BLS, particularly when dealing with smaller datasets, as it can lead to the prevalent issue of overfitting. This problem is prominent in small-scale plants where acquiring a significant amount of training data is challenging.

To address this issue, this paper proposes integrating the Dropout [15] method into C-BLS during the incremental learning process, resulting in the construction of Convolutional Broad Learning System with Dropout Incremental Learning (CDIL-BLS). This approach involves randomly hiding nodes added to the BLS model, thereby improving the accuracy and robustness of the model. The BLS component of CDIL-BLS is illustrated in Figure 4.

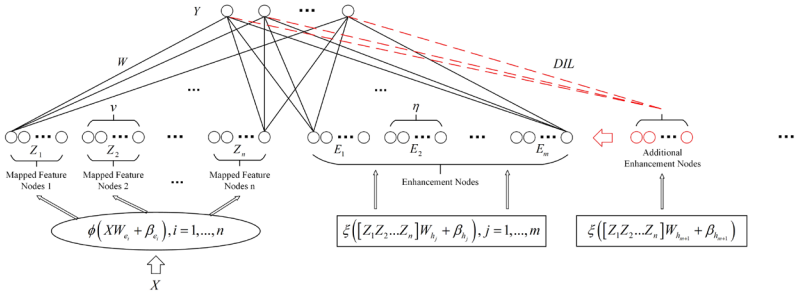

Figure 4: The BLS component of the Convolutional Broad Learning System with Dropout Incremental Learning (CDIL-BLS) model.

In the incremental learning without Dropout method, once the basic BLS model is trained, enhancement nodes are introduced by adding new columns to the matrix H. These newly added nodes are generated using the enhancement node generation method. The concatenation of these new nodes with the original nodes can be expressed as

$$H' = \left[H|\xi\left(Z^n W_{h_{m+1}} + \beta_{h_{m+1}}\right)\right] = [H|E_{m+1}] \tag{5}$$

where $W_{h_{m+1}}$ and $\beta_{h_{m+1}}$ represent the newly generated random weights and biases, respectively.

To incorporate the Dropout method, in each round of incremental learning, a $(0,1)$ vector Θ of length equal to the number of newly added enhancement nodes is generated using the Bernoulli function. This vector assigns a value of 0 to some elements and 1 to the remaining elements, with a probability of p. Subsequently, a square matrix A is created, where each row consists of the same elements as the vector Θ. The matrix E_{m+1} is then multiplied by the square matrix A to apply the Dropout mechanism. In order to preserve the mathematical expectation of the newly added nodes, it is necessary to scale the nodes after the Dropout process. Consequently, the new concatenation in the connection weight calculation is given by

$$H'_D = [H|E_{m+1} \times A \,/\, (1-p)] \tag{6}$$

At this stage, the newly generated enhancement nodes are analogous to the nodes in the upper layer of the Dropout structure. They undergo multiplication and scaling using the square matrix A to become the nodes after Dropout processing, while the subsequent layer consists of the data labels Y.

The inclusion of the Dropout method weakens the dependency between the nodes in the BLS, so that the overfitting phenomenon that occurs in incremental learning is fully suppressed. The problem of small and medium-sized plants that have difficulty in obtaining ideal training results due to the lack of sufficient training samples can also be solved using this scheme.

The next step involves calculating the new connection weights using the results obtained from the

ISSN 2616-5775 Vol. 7, Issue 2: 48-58, DOI: 10.25236/AJCIS.2024.070207

previous computations. Thanks to the intricate structure of BLS, this process requires only a few calculations to derive the updated results. It can be expressed as

$$W' = \begin{bmatrix} W - DB^{\mathrm{T}}Y \\ B^{\mathrm{T}}Y \end{bmatrix} \tag{7}$$

where

$$D = (H'_D)^{+}\xi\left(Z^n W_{h_{m+1}} + \beta_{h_{m+1}}\right) \tag{8}$$

$$B^{\mathrm{T}} = \begin{cases} (C)^{+} & , \ C \neq 0 \\ (1 + D^{\mathrm{T}}D)^{-1}B^{\mathrm{T}}(H'_D)^{+}, & C = 0 \end{cases} \tag{9}$$

$$C = \xi\left(Z^n W_{h_{m+1}} + \beta_{h_{m+1}}\right) - H'_D D \tag{10}$$

The symbol $(\,)^{+}$ represents the pseudo-inverse of the matrix.

Thus, a round of incremental learning for CDIL-BLS is completed. By continuously iterating this process, the desired model and optimal results can be obtained. The testing and incremental learning process described earlier is depicted in Figure 5.

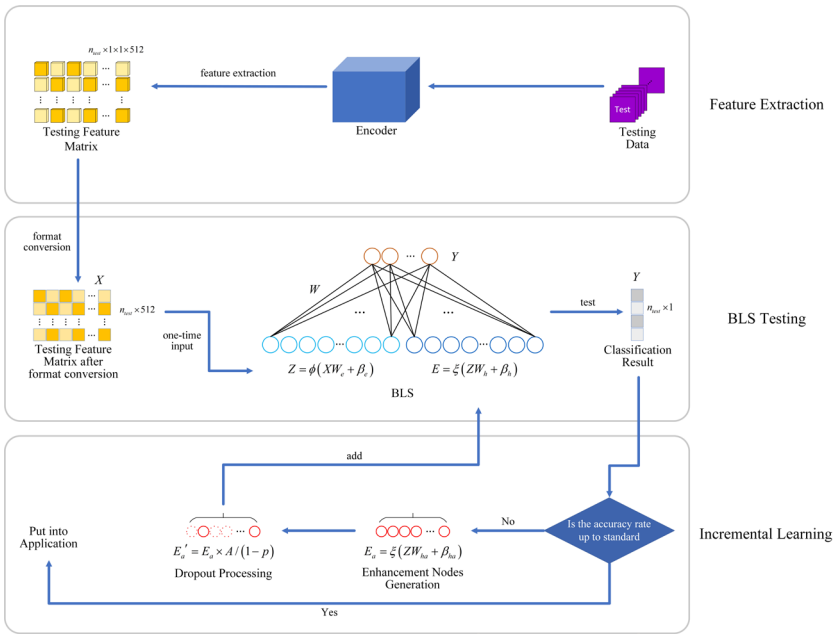

Figure 5: Testing and incremental learning process for CDIL-BLS.

4. Experiments and Analysis of Results

The experiments in this paper utilize the Keras neural network library in Python 3.10 and are conducted on the Windows 11 operating system. The CPU employed is an AMD Ryzen 7 5800H 3.20 GHz, accompanied by 16 GB of RAM. Additionally, a NVIDIA GeForce RTX 3050 Laptop GPU with 4GB VRAM is utilized.

4.1. Dataset

The experiments use NEU-CLS [16], a classification task dataset sourced from the Northeastern University surface defect database. This dataset comprises six common surface defect images found in hot-rolled steel strips, namely rolled-in scale (RS), patches (Pa), crazing (Cr), pitted surface (PS), inclusion (In), and scratches (Sc), as shown in Figure 6. The dataset consists of a total of 1800 grayscale images, with 300 samples available for each type of surface defect.

ISSN 2616-5775 Vol. 7, Issue 2: 48-58, DOI: 10.25236/AJCIS.2024.070207

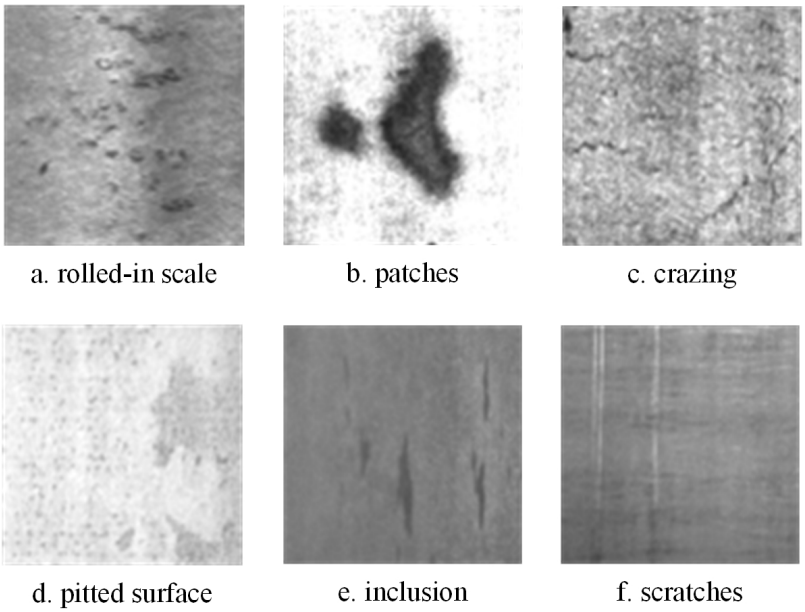

Figure 6: Types of surface defects on hot-rolled steel strip.

This dataset shows the actual situation of real-life production environments. It is important to note that different surface defect types may exhibit similar morphologies despite originating from distinct causes. Additionally, significant variations can exist within images depicting the same defect type, such as differences in depth, direction, and the number of scratches. Moreover, due to inconsistent lighting conditions in plants, the captured images display substantial variations in brightness. These factors contribute to the complexity of the dataset, posing challenges in training the model effectively.

4.2. Experiment Results

In this section, we comprehensively evaluate the superior performance of the proposed method in many aspects: effectiveness of the improvements, parameter sensitivity, comparative results with some mainstream methods, recognition effect for each type of defect, and incremental learning.

4.2.1. Ablation Experiments

To validate the effectiveness of our improvements, we conduct ablation experiments on C-BLS, comparing it with the original ResNet-18 and BLS. The training batch size of ResNet-18 is set to 32, the optimizer is stochastic gradient descent, and the learning rate is 0.001. In the case of BLS, all data is input simultaneously during both the training and testing processes. The number of mapped feature nodes, denoted as $n \times v$, is set to 10×5, while the number of initial enhancement nodes N_3 (i.e., $m \times \eta$) is set to 40. These values are selected based on subsequent discussions and evaluations. The regularization coefficient c and shrink coefficient s are 2×10^{-30} and 0.8, respectively.

Table 1: Results of ablation experiments.

Model	Accuracy (%)	Training time (s)		Training epochs	Testing time (s/sample)
ResNet-18	96.85	2.68×10^{2}		26	2.83×10^{-3}
BLS	47.50	8.32		1	2.88×10^{-4}
C-BLS	98.06	ResNet-18 initial training	1.07×10^{2}	10	2.79×10^{-3}
		BLS training	1.12×10^{-1}	1	

The accuracy, training time, training epochs, and testing time of the three models are presented in Table 1. ResNet-18 achieved its highest accuracy, reaching 96.85% after 26 epochs of training, which took 268 seconds. According to the original method, the pixel values of each image in the dataset were taken out by rows and directly input to BLS for training and testing, resulted in significantly lower accuracy of only 47.50%. In contrast, C-BLS employed 10 epochs of initial training using ResNet-18, followed by BLS training. This combined approach took a total of 108 seconds and achieved an accuracy of 98.06%. Compared to ResNet-18, C-BLS demonstrated a 59.70% reduction in training time while improving accuracy by 1.21%.

ISSN 2616-5775 Vol. 7, Issue 2: 48-58, DOI: 10.25236/AJCIS.2024.070207

The disadvantage that deep learning models tend to fall into local optima caused ResNet-18's recognition performance to improve slowly in the middle and late stages of training. On the other hand, while BLS exhibited shorter training and testing time compared to the other two models, its practical applicability was limited due to its low accuracy. We think that the pixel data extracted by rows lost crucial positional information within the images, leading to inadequate feature learning by BLS. The combination approach of C-BLS effectively addressed these issue, resulting in improved performance.

4.2.2. Parameter Sensitivity Experiments

In order to determine the optimal depth of ResNet, a crucial hyperparameter in C-BLS, ResNet-18, ResNet-34 and ResNet-50 were experimented with BLS to form C-BLS, respectively, and the results are illustrated in Figure 7. Additionally, two earlier proposed CNN models were used as comparisons in the experiments.

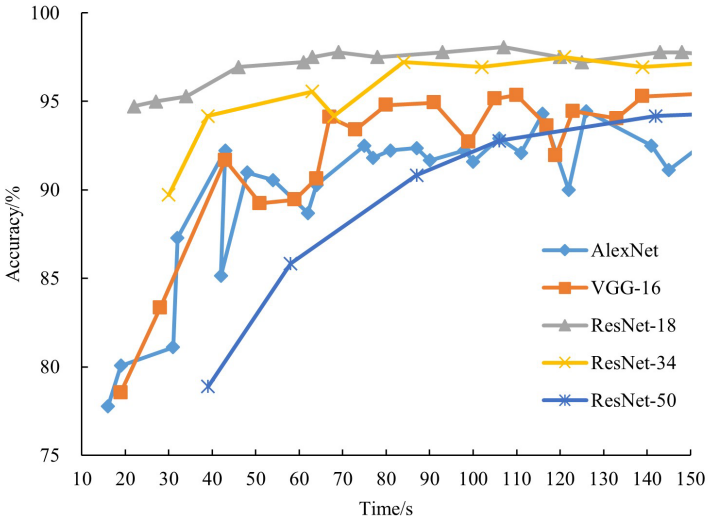

Figure 7: Experimental results of C-BLS formed by different Convolutional Neural Networks (CNNs).

The figure depicts the total training time for each C-BLS as well as the corresponding accuracies achieved through training. Each individual data point represents the C-BLS formed by a specific CNN after different epochs of training. Notably, the C-BLS formed by ResNet-18 exhibited an accuracy close to 95% after just one epoch of training. As the number of training epochs increased, the accuracy of this combination continued to improve, reaching a peak of 98.06% in the 10th epoch, with a training time of 108 seconds.

In contrast, the C-BLS formed by ResNet-34 showed comparatively lower effectiveness for a significant portion of the training period. By the 7th epoch, with a training time of 121 seconds, it achieved an accuracy of 96.94%. On the other hand, the C-BLS formed by ResNet-50 demonstrated gradually improving performance, but it required 140 seconds for 5 epochs of training, and its accuracy only reached 94.17% at that point. Furthermore, the C-BLS formed by AlexNet or Visual Geometry Group model (VGG-16) consistently struggled to surpass the 95% accuracy threshold. Considering the hardware limitations of most small and medium-sized plants and the desire to minimize model training time, ResNet-18 undoubtedly emerges as the optimal choice for constructing C-BLS.

To observe the effect of the number of mapped feature nodes ($n \times v$) and the number of initial enhancement nodes (N_3) on the accuracy in C-BLS, we conducted experiments by adjusting the parameters within the following ranges: {1×10, 2×10, 3×10, 4×10, 5×10, 6×10, 7×10} for $n \times v$, and {10, 20, 30, 40, 50, 60, 70} for N_3. The results of these experiments are presented in Figure 8. With the increase of the number of mapped feature nodes, the accuracy of C-BLS rose first and then decreased slightly. The highest accuracy was achieved when utilizing 10×5 mapped feature nodes. Similarly, with the increase of the number of initial enhancement nodes, the accuracy also tended to rise first and then decrease, roughly conforming to the normal distribution. The peak accuracy was attained when the number of initial enhancement nodes was set to 40.

ISSN 2616-5775 Vol. 7, Issue 2: 48-58, DOI: 10.25236/AJCIS.2024.070207

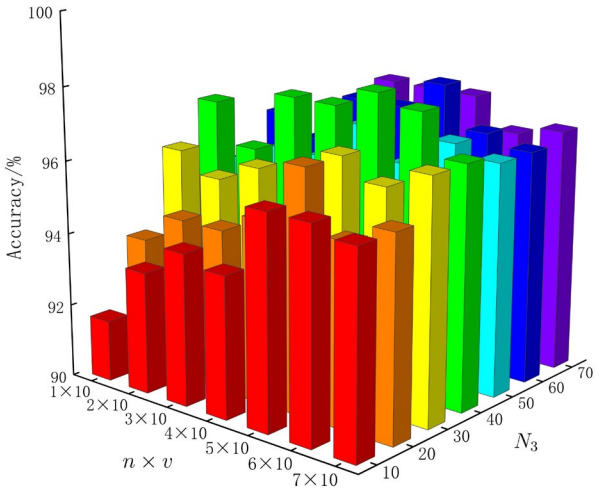

Figure 8: Effect of the number of mapped feature nodes and the number of initial enhancement nodes on the accuracy in C-BLS.

4.2.3. Comparison Experiments

We further compare the presented method with common algorithms that can be used for image recognition, including K-Nearest Neighbors (KNN) and Support Vector Machine (SVM) in machine learning, as well as AlexNet, VGG-16, GoogLeNet and Dense Convolutional Network (DenseNet-121) in deep learning.

Table 2: Results of comparison experiments.

Model	Accuracy (%)	Training time (s)		Training epochs	Testing time (s/sample)
KNN	85.76	1.00×10^{-3}		1	6.11×10^{-5}
SVM	90.61	2.79×10^{-1}		1	3.61×10^{-5}
AlexNet	97.50	2.21×10^{2}		37	7.57×10^{-3}
VGG-16	96.67	1.25×10^{3}		31	1.65×10^{-2}
GoogLeNet	97.22	3.70×10^{2}		31	1.20×10^{-2}
DenseNet-121	97.78	2.98×10^{2}		23	1.37×10^{-2}
C-BLS	98.06	ResNet-18 initial training	1.07×10^{2}	10	2.79×10^{-3}
		BLS training	1.12×10^{-1}	1	

As shown in Table 2, the accuracies of the two machine learning algorithms KNN and SVM were 85.76% and 90.61%, respectively, for the experiments using the surface defect dataset. The data of the four deep learning algorithms were selected to be counted in the table when their accuracies reached the peaks. Among them, VGG-16 took the longest time to train, much longer than the other three, and had the lowest accuracy among the four. GoogLeNet reached 97.22% accuracy after 31 epochs of training, taking 370 seconds. AlexNet's training time was the shortest among the four deep learning algorithms, totaling 221 seconds, but still 2.1 times longer than C-BLS. This algorithm achieved an accuracy of 97.50%, which was 0.56% different from C-BLS. The accuracy of DenseNet-121 is close to that of C-BLS, but its training time is still long at 298 seconds.

The training and testing time of the two machine learning algorithms were extremely short due to the relatively rudimentary computational processes involved. However, these algorithms failed to effectively capture the semantic information contained within the images, which ultimately impeded their performance breakthrough. In comparison to commonly employed deep learning algorithms, C-BLS was able to maintain the best training time and achieve a high level of accuracy. This can be attributed to its ability to avoid local optima in the later stages of deep learning and effectively feed image features into the BLS for recognition.

4.2.4. Incremental Learning Experiments

To evaluate the performance improvement of the Dropout incremental learning on C-BLS, we constructed C-BLS models using ResNet-18 and AlexNet, respectively, and conducted experiments with both the original incremental learning method and DIL. The total number of enhancement nodes in all

ISSN 2616-5775 Vol. 7, Issue 2: 48-58, DOI: 10.25236/AJCIS.2024.070207

models gradually increased from the initial 40 to 115, with an incremental addition of 5 enhancement nodes in each iteration, totaling 15 iterations. The experimental results are shown in Figure 9.

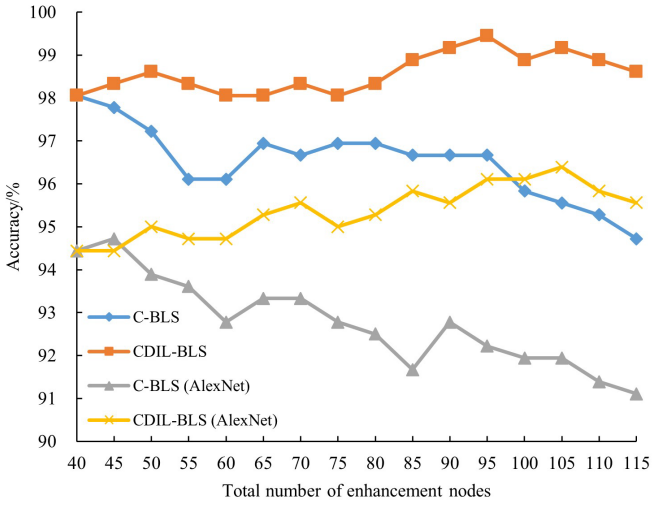

Figure 9: Accuracies during the incremental learning process.

Both CDIL-BLS models demonstrated an increasing trend in accuracy during the incremental learning process, achieving approximately a 1.5% improvement compared to the basic models at their peak performance. Notably, the CDIL-BLS model constructed with ResNet-18 achieved an accuracy of 99.44% when the total number of enhancement nodes reached 95. In contrast, the C-BLS models using the traditional incremental learning method exhibited a downward trend in accuracy, with occasional minor improvements that overall deviated from the intended direction. This discrepancy can be attributed to the overfitting issue caused by the complexity of the models. CDIL-BLS addresses this problem by utilizing the Dropout method, allowing the added enhancement nodes to effectively enhance the model's performance.

5. Conclusions

In this paper, the C-BLS is constructed to accomplish the task of classifying steel surface defects in steel production process. It achieves fast training and accurate recognition on the steel surface defect dataset with the feature extraction capability of ResNet-18 and the efficient learning capability of BLS. The optimal configuration of the CNN model and BLS node number was determined through parameter sensitivity experiments. Comparative experiments demonstrated that C-BLS met the standards for practical applications and exhibited training speeds several times faster than deep learning algorithms.

To further improve the model's performance, the CDIL-BLS is constructed by incorporating the Dropout incremental learning into C-BLS. Incremental learning in CDIL-BLS is achieved by adding enhancement nodes using the BLS flexible incremental learning algorithm combined with the Dropout method. The accuracy of CDIL-BLS gradually increased during the incremental learning process, surpassing that of C-BLS, which suffered from overfitting issues due to using the original incremental learning method.

The proposed solution in this paper ensures high model performance while minimizing the hardware resource requirements and training time consumption, making it suitable for applications in small and medium-sized plants.

References

[1] Song, G., Song, K., & Yan, Y. (2020). EDRNet: Encoder–decoder residual network for salient object detection of strip steel surface defects. IEEE Transactions on Instrumentation and Measurement, 69(12), 9709-9719.
[2] Satish, R., Murugabhoopathy, K., Rajendhiran, N., & Vijayan, V. (2020). Technology strategy for improved safety management in steel industry. Materials today: proceedings, 33, 2660-2664.
[3] Wen, X., Shan, J., He, Y., & Song, K. (2022). Steel surface defect recognition: A survey. Coatings,

ISSN 2616-5775 Vol. 7, Issue 2: 48-58, DOI: 10.25236/AJCIS.2024.070207

13(1), 17.
[4] Zhao, W., Chen, F., Huang, H., Li, D., & Cheng, W. (2021). A new steel defect detection algorithm based on deep learning. Computational Intelligence and Neuroscience, 2021, 1-13.
[5] Demir, K., Ay, M., Cavas, M., & Demir, F. (2023). Automated steel surface defect detection and classification using a new deep learning-based approach. Neural Computing and Applications, 35(11), 8389-8406.
[6] Konovalenko, I., Maruschak, P., Brezinová, J., Viňáš, J., & Brezina, J. (2020). Steel surface defect classification using deep residual neural network. Metals, 10(6), 846.
[7] Li, S., Wu, C., & Xiong, N. (2022). Hybrid architecture based on CNN and transformer for strip steel surface defect classification. Electronics, 11(8), 1200.
[8] Zhang, J., Li, S., Yan, Y., Ni, Z., & Ni, H. (2022). Surface Defect Classification of Steel Strip with Few Samples Based on Dual-Stream Neural Network. Steel research international, 93(5), 2100554.
[9] Liu, Y., Yuan, Y., Balta, C., & Liu, J. (2020). A light-weight deep-learning model with multi-scale features for steel surface defect classification. Materials, 13(20), 4629.
[10] Fu, G., Sun, P., Zhu, W., Yang, J., Cao, Y., Yang, M. Y., & Cao, Y. (2019). A deep-learning-based approach for fast and robust steel surface defects classification. Optics and Lasers in Engineering, 121, 397-405.
[11] Zhao, W., Song, K., Wang, Y., Liang, S., & Yan, Y. (2023). FaNet: Feature-aware network for few shot classification of strip steel surface defects. Measurement, 208, 112446.
[12] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).
[13] Chen, C. P., & Liu, Z. (2017). 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.
[14] Gong, X., Zhang, T., Chen, C. P., & Liu, Z. (2021). Research review for broad learning system: Algorithms, theory, and applications. IEEE Transactions on Cybernetics, 52(9), 8922-8950.
[15] Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. R. (2012). Improving neural networks by preventing co-adaptation of feature detectors. arXiv preprint arXiv:1207.0580.
[16] Bao, Y., Song, K., Liu, J., Wang, Y., Yan, Y., Yu, H., & Li, X. (2021). Triplet-graph reasoning network for few-shot metal generic surface defect segmentation. IEEE Transactions on Instrumentation and Measurement, 70, 1-11.