BLNN: Multiscale Feature Fusion-Based Bilinear Fine-Grained Convolutional Neural Network for Image Classification of Wood Knot Defects

Mingyu Gao,1 Fei Wang2,3 Peng Song4, Junyan Liu2,3 and DaWei Qi1

1College of Science, Northeast Forestry University, Harbin 150040, China
2School of Mechatronics Engineering, Harbin Institute of Technology, Harbin 150001, China
3State Key Laboratory of Robotics and System, Harbin Institute of Technology, Harbin 150001, China
4School of Instrumentation Science and Engineering, Harbin Institute of Technology, Harbin 150001, China

Correspondence should be addressed to Fei Wang; wangfeipublic@163.com

Received 11 July 2021; Revised 21 July 2021; Accepted 6 August 2021; Published 18 August 2021

1. Introduction

Wood knot defect detection is an important link in evaluating wood quality, which ultimately affects the quality of wood products [1]. Rapid detection of knot defects on wood surface can effectively improve the qualified rate of wood products [2, 3]. Consequently, it is important to identify the defects of wood knots in a short time. Although manual recognition is accurate, it takes a lot of time to train the staff, and the recognition speed on the assembly line is very slow compared to machine recognition [4, 5]. With the development of artificial intelligence and computer vision technology, deep learning has potential significance in the application of wood knot defect classification [6–8].

In recent years, image recognition based on artificial neural network and image processing has been widely studied. In order to identify the target accurately, the first step is to extract image features. For example, a Hu invariant moment feature extraction method combined with a BP (back propagation) neural network to classify wood knot defects was proposed by Qi and Mu [9]. The accuracy of this method for wood knot defect recognition is over 86%. In the same year, Khwaja et al. proposed a defect detection and classification method for wet-blue leather using artificial neural network (ANN). The features of several defects on leather were extracted by using grey level cooccurrence matrix (GLCM) and grey level run-length matrix (GLRLM). The acquired features are passed to the multilayer perceptron using the Levenberg-Marquardt (LM) algorithm. The accuracy of this model is 97.85% [10]. In 2021, Aditya et al. proposed a method based on statistical texture features in GLCM to classify leaf blight of four plants by selecting appropriate thresholds. The accuracy of this method can reach 74% under optimal conditions [11]. The above methods require manual feature extraction, and the recognition rate is not high. Consequently, a convolutional neural network (CNN) which can
automatically learn the target features is needed to replace the complex artificial defect feature extraction. In 2020, Zhang et al. proposed a CNN image recognition algorithm for supermarket shopping robots. This algorithm overcomes the problems of low accuracy and slow speed in image recognition. The experimental results show that the accuracy of the algorithm can reach more than 98%. It also verifies that the image recognition algorithm can be applied to supermarket shopping robots to meet the needs of competition [12]. In the same year, Liu et al. proposed an intangible cultural heritage image recognition model based on color feature extraction and CNN, with the recognition rate reaching 94.8% [13]. In 2021, a new method based on transfer learning and ResNet-34 convolutional neural network for recognizing wood knot defects was presented by Gao et al. The experimental results show that the classification accuracy of this method can reach 98.69% [14]. Although these methods are practical, their accuracy can still be improved, and they have less application in wood knot defect detection. In order to solve these problems, improve the accuracy and recognition
Figure 2: Continued.
speed of the model, and reduce the training time, a high-accuracy wood knot defect detection method based on convolutional neural network is required.

In this paper, a bilinear classification model based on feature fine-grained fusion strategy named BLNN was proposed to detect wood knot defects. This paper is arranged and structured as follows. Firstly, the dataset of wood knot defects is acquired and augmented. Then, the proposed BLNN model is introduced. Subsequently, the network is trained and tested by using the dataset of wood knot defects. Finally, based on a benchmark dataset, the test results are compared and analyzed with other deep learning models.

2. Materials and Methodology

2.1. Dataset Acquisition. The dataset was downloaded from the website of the Computer Laboratory, Department of Electrical Engineering, University of Oulu [15–17], and

Figure 2: Original images of wood knot defect and those created through data augmentation: (a) original image, (b) vertical mirroring, (c) rotation by 180°, (d) horizontal mirroring, (e) adding Gaussian noise to image, (f) increasing the hue by 10, and (g) adding salt-and-pepper noise to the image.
consists of 365 images with four types of spruce knot defects. These are dry knot, edge knot, leaf knot, and sound knot, respectively. Figure 1 shows the four types of wood knot defects in the dataset used in this paper.

2.2. Image Preprocessing and Augmentation. Deep learning networks have to be trained on massive datasets to achieve good performance [18]. Therefore, when the original dataset contains a limited number of images, data augmentation [19] is required to improve accuracy and prevent overfitting [20]. In this case, six methods are employed to augment the dataset, namely, vertical mirroring, rotation by 180°, horizontal mirroring, adding Gaussian noise, increasing the hue by 10, and adding salt-and-pepper noise. Consequently, the number of images was increased to seven times the original number. Due to more image augmentation, the learning ability of the network has increased. The data augmentation is shown in Figure 2. Table 1 lists the names and the number of images used for the experiments. Eventually, the dataset was randomly divided into a training set, a validation set, and a testing set in ratio of 3:1:1.

2.3. Proposed Classification Model. A CNN network called BLNN is proposed for fine-grained feature extraction [21–23] based on images, which consists of two different branching convolutional neural networks. Since the two CNNs are different, they are used to extract features of different scales. These two features are confluence together to form a one-dimensional feature vector using the bilinear pooling operation [24, 25], and finally, the feature vector is classified using a classifier to obtain the recognized class. An overview of the proposed network architecture is shown in Figure 3. The parameters of BLNN are shown in Table 2.

2.4. Multiscale Information Fusion Strategy. The core of the BLNN lies in the fusion of two bilinear layer output vectors. According to this, a CNN-based fusion network structure is proposed to extract information about wood knot defects from different dimensions. BLNN can be expressed as follows:

$$ B = (F_1, F_2, F_{c1}, F_{c2}), $$

where $F_1$ and $F_2$ denote two feature extraction functions and $F_{c1}$ and $F_{c2}$ are the fully connected layers.

$$ F_1 = (C, B, R, P, F_{c11}), $$

$$ F_2 = (C, R, P, F_{c21}), $$

where $C$, $B$, $R$, $P$, $F_{c11}$, and $F_{c21}$ denote the convolutional layer [26], BatchNorm layer [27], ReLU activation function...
and fully connected layers of $F_1$ and $F_2$, respectively. First of all, the algorithm uses two branch networks named $F_1$ and $F_2$ to train the wood knot defect images, respectively. A smaller $3 \times 3$ convolutional kernel is used in $F_1$ to extract a rough feature; it can reduce the parameters. $F_2$ uses a larger $8 \times 8$ convolutional kernel...
to extract features. The larger convolutional kernel can provide higher receptive field and extract more fine features. Therefore, \( F_2 \) is designed to capture the fine-grained characteristics [31] of wood knots. The fusion of two branches in the fully connected layer is shown in Figure 4.

After the first fully connected layer, vectors \( x_1 \) and \( x_2 \) with a dimension of \( 1 \times 120 \) are obtained from the two branches, respectively (Figure 4). Then, \( x_1 \) and \( x_2 \) cascade to get \( x_3 \). Cascade fusion [32] is employed to superpose the two outputs, which can be expressed as follows:

\[
f_{\text{cas}}(x_1, x_2),
\]

where \( x_1 \) and \( x_2 \) are the outputs behind \( F_{c11} \) and \( F_{c21} \), respectively. Two vectors are cascaded and spliced along the vertical axis into one vector with a dimension of \( 1 \times 240 \). Therefore, the vector \( x_3 \) contains all the eigenvectors computed by the two branches, which is computed from the image features of two different scales, and the features are represented more comprehensively. Next, a one-dimensional vector with a dimension of \( 1 \times 50 \) is set after \( x_3 \), and finally, set the output of the fully connected layer to 4, indicating the category of classification.

2.5. Loss Function and Optimizer. The loss function is applied to evaluate the difference between the predicted and actual values of the model [33–35]. The smaller the difference, the smaller the cross-entropy. This study uses the cross-entropy loss function, which is expressed as follows:

\[
L = - \sum_{i=1}^{n} p_i(x) \log [q_i(x)],
\]

where \( L \) represents the loss value of the sample and \( p_i(x) \) and \( q_i(x) \) represent the target output and the actual output, respectively. Cross-entropy overcomes the problem that weights and deviations are updated too slowly. When the error is large, the weight updates quickly, and when the error is small, the weight updates slowly.

The optimizer is used to update and compute the network parameters that affect the model training and output to approximate or reach the optimal value, thereupon then minimizing (or maximizing) the loss function [36]. In this case, the Adam optimizer is used. The Adam optimizer combines the advantages of AdaGrad [37] and RMSProp [38]. It takes the first-order moment estimation (i.e., the mean of the gradient) and second-order moment estimation (i.e., the uncentered variance of the gradient) of the gradient into account and calculates the update step. Adam is simple to implement, is computationally efficient, and has low memory requirements, and the hyperparameters usually require no or little fine-tuning.

3. Experiment Results and Discussion

The experiment was performed on a Windows 10 64-bit PC equipped with an Intel(R) Xeon(R) Bronze 3204 CPU @ 1.90 GHz processor and 128 GB RAM. The deep learning programs were run on two NVIDIA GeForce RTX 3090 GPUs with 24 G RAM. The code is mainly implemented in Python, including data preprocessing and algorithm implementation. The deep learning framework is Pytorch. The experimental environment is shown in Table 3.

<table>
<thead>
<tr>
<th>Hardware environment</th>
<th>Software environment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memory</td>
<td>128.00 GB</td>
</tr>
<tr>
<td>CPU</td>
<td>Intel(R) Xeon(R) Bronze 3204 CPU @ 1.90 GHz (6 core)</td>
</tr>
<tr>
<td>Graphics card</td>
<td>NVIDIA GeForce RTX 3090 (24 G)</td>
</tr>
</tbody>
</table>

### 3.1. Model Training

In this study, the dataset is divided into a training set, a validation set, and a testing set, which contain 1534, 518, and 503 images, respectively. The hyperparameter setting for model training is shown in Table 4. The epoch, batch size, and learning rate are set to 200, 128, and 1e−3 to make all models converge stably. The model training process is shown in Figure 5.

<table>
<thead>
<tr>
<th>Related parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Batch size</td>
<td>128</td>
</tr>
<tr>
<td>Learning rate</td>
<td>1e−3</td>
</tr>
<tr>
<td>Epoch</td>
<td>200</td>
</tr>
<tr>
<td>Optimizer</td>
<td>Adam</td>
</tr>
<tr>
<td>Loss function</td>
<td>Cross-entropy</td>
</tr>
<tr>
<td>CUDA</td>
<td>Enable</td>
</tr>
<tr>
<td>CUDNN</td>
<td>Enable</td>
</tr>
</tbody>
</table>

3.1.1. The Training Results of the BLNN Model. The accuracy and loss curves for the training and verification stages are shown in Figure 6, respectively.

Figure 6 shows that the model has trained 200 epochs; it can be seen that the training accuracy of the model remains stable after 50 epochs. Most of the fluctuations are between 0.95 and 1.00, and the loss decreases to around 0.2 to 0.35 with little fluctuation. After nearly 100 epochs, the loss of training phase decreased to about 0.2, but there are still fluctuations. The accuracy remained stable during the validation phase, most of which fluctuated between 0.95 and 1.00. Better classification results are obtained.
3.1.2. Contrast Experiment. The results of BLNN are compared with those of AlexNet, VGGNet-16, GoogLeNet, ResNet-18, and MobileNet-V2 to verify the effectiveness of the model. ResNet-18 achieves feature reuse by identity shortcut. Similar to ResNet, the fusion strategy of BLNN is to combine in-depth and shallow-depth features to obtain more detailed feature information. By comparing the performance of different network structures on the same wood knot defect dataset, the effectiveness and the superiority in identifying wood knot defects of BLNN are proved.

As shown in Figure 7, BLNN has a faster convergence rate than other models and finishes convergence at the 50th epoch. Consequently, a smaller epoch has the opportunity to be chosen to use in practice.

Five learning rates, 0.1, 0.01, 0.001, 0.0001, and 0.00001, were tested after establishing the BLNN model. The experimental results are shown in Table 5.

In Table 5, it is observed that when the learning rate is 0.1, the model does not converge effectively. The main reason is that an excessively large learning rate will cause the parameters of the model to oscillate beyond the valid range rapidly. When the learning rate has been reduced to 0.01, 0.001, and 0.0001, good results have been achieved, the error has been converged, and test accuracy has reached 94.43%, 99.20%, and 96.62%, respectively. When the learning rate continues to drop to 0.00001, the network convergence is very slow and the time to find the optimal value increases. At the same time, convergence may occur when entering the local extreme point, and no optimal value can be found. By continuously reducing the learning rate, it is found that the training results of different learning rates are different. Consequently, considering the accuracy and training time of the model, 0.001 is chosen as the initial learning rate to train the model.
The optimization algorithm is applied to find the optimal solution of the model. In this case, the Adam is employed and compared with SGD, AdaGrad, and Adax, as shown in Figure 8. The results show that the model with Adam has the fastest convergence speed and the highest accuracy. Table 6 shows the prediction results of the four optimization algorithms under the same condition. The results show that the accuracy of SGD, AdaGrad, Adamax, and Adam is 79.32%, 94.04%, 98.01%, and 99.20%, respectively. Consequently, considering the accuracy and training time of the model, Adam is chosen as the optimizer of the model.

### 3.2. Evaluation Metrics

To evaluate the performance of the BLNN, the precision \( P \), recall \( R \), F1 score \( F_1 \), and false alarm rate (FAR) were applied for the evaluation shown as follows:

\[
P = \frac{TP}{TP + FP}, \tag{5a}
\]

\[
R = \frac{TP}{TP + FN}, \tag{5b}
\]

\[
FAR = \frac{FP}{FP + TN}. \tag{5c}
\]

The optimization algorithm is applied to find the optimal solution of the model. In this case, the Adam is employed and compared with SGD, AdaGrad, and Adax, as shown in Figure 8. The results show that the model with Adam has the fastest convergence speed and the highest accuracy. Table 6 shows the prediction results of the four optimization algorithms under the same condition. The results show that the accuracy of SGD, AdaGrad, Adamax, and Adam is 79.32%, 94.04%, 98.01%, and 99.20%, respectively. Consequently, considering the accuracy and training time of the model, Adam is chosen as the optimizer of the model.

### Table 5: The comparison of results in different learning rates.

<table>
<thead>
<tr>
<th>Learning rate</th>
<th>Number</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>250</td>
<td>49.70</td>
</tr>
<tr>
<td>0.01</td>
<td>475</td>
<td>94.43</td>
</tr>
<tr>
<td>0.001</td>
<td>499</td>
<td>99.20</td>
</tr>
<tr>
<td>0.0001</td>
<td>486</td>
<td>96.62</td>
</tr>
<tr>
<td>0.00001</td>
<td>436</td>
<td>86.68</td>
</tr>
</tbody>
</table>

### Figure 7: Results in the training set of all the applied models.

(a) Train loss

(b) Val accuracy

---

Journal of Sensors
\[ \text{FP} \cdot \text{TP} + \text{FP}, \quad (5d) \]

where TP, FP, TN, and FN represent the true positive, false positive, true negative, and false negative.

3.3. Model Evaluation. The performance of BLNN is evaluated in the task of wood knot defect classification. 503 wood knot defect images were used as testing dataset. The trained BLNN was compared with AlexNet, GoogLeNet, MobileNet, ResNet-18, and VGGNet-16, and the network was evaluated according to confusion matrix, precision, recall, \( F1 \) score, FAR, accuracy, training time, and detection time.

As shown in the confusion matrix in Figure 9, the accuracy of each category is described by comparing the actual category with the predicted category. The numerical
distribution of confusion matrix shows that AlexNet and BLNN have better classification results. BLNN can recognize edge knot and sound knot up to 100%, and dry knot and leaf knot are slightly lower than AlexNet, which is the direction to improve in the future. However, as shown in Figure 10, BLNN has the highest overall recognition rate of knot defects, reaching 99.20%. Table 7 shows the training time and the detection time of all models for each wood image. It can be seen that BLNN has the shortest training time and the fastest detection speed in all models due to its fewer parameters and higher feature extraction ability.

Precision, recall, $F_1$, and FAR of the four categories of wood knot defect images in the testing set are shown in Figure 11. It can be seen that BLNN is superior to MobileNet-V2, ResNet-18, and VGGNet-16 in the classification of four wood knot defects. Compared with AlexNet and GoogLeNet, some of the BLNN metrics are slightly worse, but the gap is not big, which requires further improvement in the future. As shown in Figure 10 and Table 7, although BLNN is not always optimal in these models, BLNN has the highest accuracy and the fastest training time and detection speed, and it is easy to be built and embedded into other models because of its small parameters and computation, which makes it possible to identify wood knot defects. Compared with other models, BLNN has obvious advantages in accuracy and calculation, so it has more practical application value. An unexpected phenomenon is that MobileNet,
ResNet-18, and VGGNet-16 do not achieve the desired performance, especially ResNet which has the lowest recognition rate. Therefore, the network structure has a great impact on the training results.

As shown in Figure 3, BLNN consists of two single-branch networks. To verify the improvement of model performance by using two-branch networks, the upper and lower branches of BLNN are compared with BLNN, respectively. The results are shown in Figures 12 and 13.

From Figures 12 and 13, it can be seen that BLNN has the fastest convergence speed and highest accuracy in the three networks. In addition, the convergence speed of the upper branch network in the training set is faster than that of the lower branch network, and the performance of the lower branch network in the verification set is better than that of the upper branch network. As shown in Figure 13, BLNN has the best performance, the lower network has the second performance, and the upper network has the worst performance, because the upper network uses $3 \times 3$ convolutional kernel, the lower network uses $8 \times 8$ convolutional kernel,

### Table 7: Training time and detection time of all the applied methods.

<table>
<thead>
<tr>
<th>Method</th>
<th>Training time (min)</th>
<th>Detection time (s/image)</th>
</tr>
</thead>
<tbody>
<tr>
<td>AlexNet</td>
<td>37.32</td>
<td>0.2744</td>
</tr>
<tr>
<td>GoogLeNet</td>
<td>44.27</td>
<td>0.3519</td>
</tr>
<tr>
<td>MobileNet-V2</td>
<td>12.97</td>
<td>0.2425</td>
</tr>
<tr>
<td>ResNet-18</td>
<td>15.95</td>
<td>0.4573</td>
</tr>
<tr>
<td>VGGNet-16</td>
<td>36.88</td>
<td>1.9583</td>
</tr>
<tr>
<td>BLNN</td>
<td>11.22</td>
<td>0.0795</td>
</tr>
</tbody>
</table>

![Figure 10](image-url)
and the lower network has a larger receptive field. Therefore, the bilinear structure of BLNN has better performance than that of single-branch networks.

As shown in Figure 3, BLNN has two single-branch networks. The upper and lower branch networks use different sizes of convolutional kernel; the upper branch network convolutional kernel is $3 \times 3$, and the lower branch network convolutional kernel is $8 \times 8$. To verify the effect of different convolutional kernel sizes on the model performance, we separately use BLNN (the upper branch network is $3 \times 3$, the lower branch network is $8 \times 8$) compared with two networks with $3 \times 3$ and $8 \times 8$; the results are shown in Figures 14 and 15.

From Figures 14 and 15, it can be seen that BLNN has the fastest convergence speed and highest accuracy in these three networks. In addition, the network with convolutional kernel size $3 \times 3$ in the training set converges faster than $8 \times 8$, and the network with convolutional kernel size $8 \times 8$ in the verification set performs better than $3 \times 3$. As shown in Figure 15, BLNN performs best, the network with convolutional kernel size $8 \times 8$ performs second, and the network with convolutional kernel size $3 \times 3$ performs worst. This is because networks with $8 \times 8$ convolutional kernel have a larger receptive field, but BLNN uses dual-branch networks with different sizes of convolutional kernel, smaller convolutional kernel ($3 \times 3$) for upper branch networks to extract local
details and larger convolutional kernel (8 × 8) for lower branch networks to extract more comprehensive global information, and then, these two kinds of feature information are fused. More comprehensive information can be acquired, so the performance of BLNN is better than that of the other two networks with different convolutional kernels.

3.4. Model Generalization. In order to evaluate the generalization ability of BLNN, we tested the classification ability of BLNN on some boards. Green means correct recognition was used to mark in green and the wrong recognition was marked in grey in this case. Details of the identification such as the name and probability of wood knot defects are displayed next to each label. Figure 16 shows four wood knot defects and the corresponding identification results.

It can be seen that most of the wood knot defects in the image are correctly identified. Some of the wood knot defects are similar in shape to other defects, and some of the wood defects are not trained, which makes the model appear to identify errors. In most cases, our method (BLNN) still has high accuracy. This indicates that BLNN has certain application value in practice.

As shown in Figure 16, since we only focus on the four defects of dry knot, edge knot, leaf knot, and sound knot.
when training the network, it can be seen that there are some defects that have not been identified. This is one of our future research directions to increase the types of defect classification.

3.5. Discussion. The effectiveness of BLNN can be discussed in two aspects.

3.5.1. Feasibility of Bilinear Network Structure. Compared with single-branch network, BLNN has obvious advantages in accuracy and convergence speed, which proves that the classification ability of the network can be improved by extracting and fusing features from the bilinear network. This network extracts features from two parallel single-branch networks, which can make the extracted features more comprehensive. This is the key to improve the classification performance. Although classical network structures such as ResNet are generally single-branch networks, their features are relatively single. Bilinear network can extract more information than a single network.

3.5.2. Rationality of Using Different Convolutional Kernel Sizes. Compared with other classical networks, BLNN has obvious advantages in accuracy and computation, which proves that the classification ability of networks can be improved by fusing local features (convolutional kernel size $3 \times 3$) and global features (convolutional kernel size $8 \times 8$) through a bilinear fusion structure. The network uses convolutional kernel with different sizes to extract multiscale features.
features from the same image, and this fine-grained information is the key to classification.

For the proposed BLNN network, the local and global features extracted by the convolutional layer are fused in the fully connected layer. In other words, it fuses all the features of different scales together through a fusion operation. Therefore, BLNN expands the number of features without generating many complex feature maps. In the fully connected layer, we improve the robustness and classification accuracy of the network by setting an appropriate number of neurons.

BLNN performs well in the classification of wood knot defects. However, performing network fusion operations in the fully connected layer may not be optimal for other tasks. This requires more research in the future.

4. Conclusion

In conclusion, a bilinear classification model based on feature fine-grained fusion strategy named BLNN was proposed in this case. The convolutional kernel size of the upper branch network of BLNN was set to $3 \times 3$, and the convolutional kernel size of the lower branch network was set to $8 \times 8$. Two different sizes of convolutional kernels were used to extract features at different scales, and feature fusion was used to classify the wood knot defects. 2052 images of wood knot defects were used for training after 200 training epochs. The experimental results show that the accuracy of BLNN reaches $99.20\%$ during the testing phase. In addition, when wood knot defects are detected by this method, a large number of image preprocessing and manual feature extraction are not demanded, which greatly improves the recognition efficiency. The speed of defect detection is only $0.0795$ s/image, and the training time is reduced. This means that BLNN has potential application value in wood nondestructive testing and wood knot defect detection and provides a feasible solution for future wood knot defect identification. In addition, the experimental results also show that multiscale information fusion is effective to improve model performance through network fusion.
Data Availability

The datasets, codes, and weight files used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the Foundation for Innovative Research Groups of the National Nature Science Foundation of China under Grant No. 51521003, NSFC under Contract No. 61571153 and No. 51173034, China National Postdoctoral Program for Innovative Talents (Grant No. BX2021092), China Postdoctoral Science Foundation (Grant No. 2021M690841), Heilongjiang Postdoctoral Fund (Grant No. LBH-Z20019), Aeronautical Science Foundation of China (No. 2020Z057077001), and Self-Planned Task of State Key Laboratory of Robotics and System (HIT), the Programme of Introducing Talents of Discipline of Universities (Grant No. B07108).

References


