Ecological Informatics 58 (2020) 101093

Contents lists available at ScienceDirect

Ecological Informatics

journal homepage: www.elsevier.com/locate/ecolinf

A biological image classification method based on improved CNN

Jiaohua Qin, Wenyan Pan, Xuyu Xiang*, Yun Tan, Guimin Hou

College of Computer Science and Information Technology, Central South University of Forestry &Technology, Changsha 410004, China

ARTICLE INFO

Keywords:
CNNs
Inverted residual block
Biological image classification

ABSTRACT

With the increase of biological images, how to classify them effectively is a challenging problem, the Convolutional Neural Networks (CNNs) show promise for this problem. The challenges of using CNNs to handle images classification lie in two aspects: (1) How to further improve the classification accuracy? (2) How to make the network more light weight? To address the above challenges, this paper proposed a biological image classification method based on improved CNN. In this paper, fixed size images as input of CNN are replaced with appropriately large size images and some modules were replaced with an Inverted Residual Block module with fewer computational cost and parameters. The proposed method extensively evaluated the computational cost and classification accuracy on five well known benchmark datasets, and the results demonstrate that compared with existing image classification methods, proposed method shows better performance image classification and reduces the network parameters and computational cost.

1. Introduction

With the improvement of image acquisition ability, a large number of biological images need rapid identification and recognition. At present, many important ecological research problems depend on image, so it is of great significance to study image classification algorithms for biological research. Some biological image are fine-grained images, such as different species of birds (Wah et al., 2010), flowers (Nilsback and Zisserman, 2008). It is a highly challenging task due to these images highly similar subordinate categories, so the gap between categories is small. There are many interference factors such as posture, angle of view and occlusion.

In recent years, Convolutional Neural Networks (CNNs) has thrived and made significant breakthroughs in the field of computer vision, including research on Target detection (Sangineto et al., 2019; Tian et al., 2017), Semantic segmentation (Long et al., 2015; Noh et al., 2015), Image classification (Chan et al., 2015; Jiao and Liu, n.d.; Zhang et al., 2019), other Recognition tasks (Yuan et al., 2015; Yuan et al., 2019) and other fields (Liu et al., 2020; Luo et al., 2019). Compared with traditional image processing methods (Ma et al., 2019), the CNNs has higher complexity and can extract higher-level image information.

Although the CNNs has made convincing achievements in many fields, the efficiency problem is an urgent problem to be solved. The efficiency problem can be divided into model storage and model prediction speed problems. CNNs contains a large number of weight parameters, which require a large amount of memory to be stored; The prediction speed must be improved if CNN is used in practice.

The early CNNs, such as AlexNet (Krizhevsky et al., 2012a) and VGGNet (Krizhevsky et al., 2012), had a great performance, this comes at a high cost: training and evaluating the network requires a lot of computation. In order to solve the efficiency problem, later networks, such as Inception (Simonyan and Zisserman, 2016), ResNet (He et al., 2016), and DenseNet (Huang et al., 2017), used different methods to reduce the network parameters. The common solution is to compress the trained model. Different from the compress of the trained model, the main idea of lightweight networks is to design more efficient network computing methods to reduce network parameters and computational cost. Therefore, there are some efficient networks emerged, such as MobileNet (Howard et al., 2017) and ShufflfleNet (Zhang et al., 2018).

Current CNNs require a fixed input image size, such as 224 × 224 or 299 × 299. For input images of arbitrary size, most existing methods are to fit the input images to fixed size by cropping (Pan and Yang, 2010) or warping (He et al., n.d.). However, cropping may lose important contents of the image, while warping may cause unnecessary geometric distortion of the image. Classification accuracy will be affected due to content loss or distortion. Appropriately increasing the input image size can effectively save the features of image and improve the classification performance. However, too large image size is used as the network input, and it will inevitably increase the network computational cost.

The most significant characteristics of CNNs are that they can learn

* Corresponding author.
E-mail address: xyuxiang@163.com (X. Xiang).

https://doi.org/10.1016/j.ecoinf.2020.101093
Received 13 January 2020; Received in revised form 11 April 2020; Accepted 11 April 2020
Available online 21 April 2020
1574-9541/ © 2020 Published by Elsevier B.V.

knowledge from one task and transfer it to another separate task. For the small dataset, fine-tuning is an effective way to train the network. The weight of the pre-training model trained on the large dataset is used as the initial weight and then use small datasets to train the network to update weights. This method has been proven to be an effective way to realize their classification tasks (Tajbakhsh et al., 2016; Zeiler and Fergus, 2014).

The training process of CNNs is to optimize the cost function. Training a CNN model may require a lot of computing resources. By using optimization algorithm, the training time can be saved to accelerate the convergence of the model. The commonly used optimization algorithms are gradient descent algorithm (Ruder, 2016), adaptive motion estimation (Kingma and Ba, 2014), and other optimization algorithms (Sun et al., 2018).

In order to improve the accuracy of biological image classification, in this paper, we use the maximum of average length and width of the image in the dataset as input to the network, this can save the effective features of image. To solve the problem of the increase of network computational cost caused by the increase of network input image size, the Inverted Residual Block in MobileNetV2 (Sandler et al., 2018) is used to replace some modules in CNN. In this way can improve not only classification accuracy but also reduce the network parameters and computational cost.

2. Related works

Before deep learning, computer vision technology is often used to process biological images (Balfoort et al., 1992; Simpson et al., 1991). With the continuous development of the deep learning, CNNs has shown a strong ability in the field of biological image processing (Kumar et al., 2016; Schneider et al., 2018). More and more researches have applied the CNNs to the field of biological image processing and achieved satisfactory results.

Early CNNs, such as AlexNet (Krizhevsky et al., 2012a) and VGGNet (Krizhevsky et al., 2012), had strict rules about the size of input images. This is because these networks consist mainly of two parts, the convolution layer, and the full connection layer. Convolution layer does not need a fixed image size and can generate a feature map of any size. The input of the full connection layer is the result of the feature mapping vectorization of the last convolution layer, and the input size is fixed. Therefore, the fixed-size constraint only comes from the full connection layer.

Current CNNs, such as Inception (Simonyan and Zisserman, 2016), ResNet (He et al., 2016), and DenseNet (Huang et al., 2017), all abandon the full connection layer and adopt Global Average Pooling layer (Lin et al., 2013). The advantage of Global Average Pooling layer is not only to solve the problem of a large number of parameters in the full connection layer, but also to realize image input of any size. References (Zheng et al., 2016) shows that appropriate increase of image size can improve classification accuracy, However, as the size of the input image increases, the amount of computation required increases by square, which is a costly method to obtain accuracy.

InceptionV3 (Simonyan and Zisserman, 2016) primarily improves the traditional convolution layer in the network, to reduce parameters while increasing network depth and width. This network mainly uses the InceptionV3 module, which the main idea is factorizing. The large convolution kernel is factorized into small convolution kernels, which can not only accelerate the calculation but also split one convolution kernel into multiple, further increasing the network depth and width. The most important function of this module is to reduce the network computational cost.

DenseNet (Huang et al., 2017) is a CNN with dense connections. In this network, the input of each layer of the network includes the output of all previous layers. In this way, the gradient vanishing problem is alleviated and the feature propagation is strengthened. These features enable DenseNet to achieve better performance with less parameters and calculation costs. Based on the advantages of the above two networks, this paper uses Inceptionv3 and DenseNet as the baseline network.

How to make the CNNs more efficient has become a hot topic for researchers, for the standard convolution, we assume that the size of the input feature map is $M_{in} \times M_{in} \times C_{in}$, the size of the input feature map is $M_{out} \times M_{out} \times C_{out}$ and the size of convolution the kernel is $K \times K$, so the computational cost C_{std} of standard convolutions is computed in Eq. (1).

$$C_{std} = K \times K \times C_{in} \times C_{out} \times M_{out} \times M_{out} \tag{1}$$

The parameters P_{std} of standard convolutions are computed in Eq. (2).

$$P_{std} = K \times K \times C_{in} \times C_{out} \tag{2}$$

where C_{in}, C_{out} are the channel of the input and output feature map, respectively.

There are many ways to make network more efficient. The main idea of network pruning (Mao et al., 2017) is to remove the relatively unimportant weights in the weight matrix, and then fine-tune the network again. Generally, parameters of neural network models are represented by floating point numbers with a length of 32 bit. In fact, there is no need to retain such high digits. Network quantization (Mohammad et al., 2016) can be adopted to reduce the space occupied by each weight by sacrificing precision. The knowledge distillation (Hinton et al., 2015) use transfer learning to train another simple network by using the output of a pre-trained complex model as a supervisory signal.

All of the above methods are to compress existing networks and build new small networks. A different approach for obtaining small networks is to use convolution with less computational cost and parameters, such as Depthwise Separable Convolution. Depthwise Separable convolution separates traditional convolution into two steps: One Depthwise convolution and 1 × 1 Pointwise convolution. Depthwise convolution convolved each channel, that is, a convolution kernel is responsible for a channel, then get the feature map of each channel. These feature maps are combined by pointwise convolution. In the standard convolutions layer, feature extraction and feature fusion should be carried out at the same time, from the perspective of the use of parameters, the efficiency is low and the performance is not excellent. Instead, Depthwise Separable convolution makes the different channels independent of each other. Feature extraction is carried out first, and then feature fusion is carried out. In this way, model parameters can be fully utilized for representation learning, and fewer parameters can be used to achieve better results. This idea has been used to design efficient deep networks, such as MovileNet (Howard et al., 2017), Xception (Chollet, 2017). MobileNetV2 (Sandler et al., 2018) proposes Inverted Residual Block, which improved the Depthwise separable convolution, make the network achieve better performance.

During the training, network parameters are randomly initialized. Fine-tuning allows us to conduct specific training according to our classification and identification tasks based on parameters trained on ImageNet datasets. Fine-tuning significantly improves the adaptability and can use smaller datasets to achieve higher classification results (Pan et al., 2019; Pan et al., 2019; Wang et al., 2018; Wang et al., n.d.).

3. Proposed method

3.1. Expand network input images size

Before training the network, the input image is adjusted to the same size. For example, for InceptionV3, during CNN training, all images are adjusted to 299 × 299 before being sent to the network. During the test, the image is usually adjusted to the same size for feature extraction and classification.

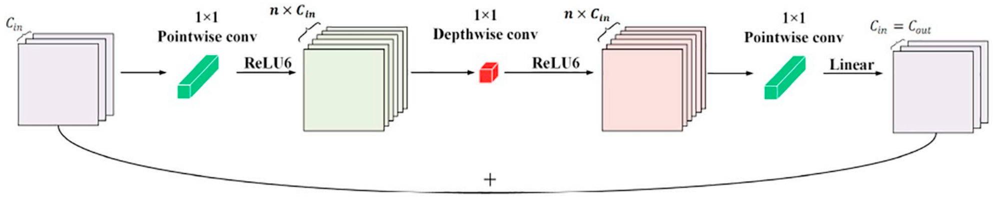


Fig. 1. Inverted residual block.

For image classification, the region of interest may only occupy a small part of the whole image, and cropping image may lose important information of the image. However, warping will change the aspect ratio of the image and change the shape of the region of interest. These will affect the classification accuracy. In larger images, details can be observed more clearly, which can effectively improve the accuracy.

Theoretically, the preferable classification accuracy can be achieved by inputting the original image into the network without any processing, but this will make the network model has large computational cost. According to Eq. (1), the larger the size of the input image (the size of the input feature map), the larger the computational cost of the network, and the longer the training and testing time. So, in this article, we choose a compromise, to avoid the training and testing process of the network too slow.

In this paper, to alleviate the impact of information loss, we use relatively large images as input. For a dataset, we first calculate the average length and width of all images in the dataset and then select the maximum value as the size of the input image. Detailed information is provided in Section 4.1.

3.2. The inverted residual block

The main idea of Inverted Residual Block is Depthwise separable convolution, which can effectively reduce the number of parameters and computation. The computational cost of Depthwise separable convolution is the sum of the calculation amount of deep convolution and 1 × 1 convolution. The computational cost C_{dsc} of Depthwise separable convolution is shown in Eq. (3):

$$C_{dsc} = K \times K \times C_{in} \times M_{out} \times M_{out} + C_{in} \times M_{out} \times M_{out} \times C_{out} \tag{3}$$

The parameters P_{dsc} of Depthwise separable are computed in Eq. (4).

$$P_{dsc} = K \times K \times C_{in} + 1 \times 1 \times C_{in} \times C_{out} \tag{4}$$

where $M_{in} \times M_{in} \times C_{in}$, the size of the input feature map, $M_{out} \times M_{out} \times C_{out}$ is the size of the output feature map and $K \times K$ is the size of convolution kernel. The computational cost ratio r_c of the Depthwise separable convolution to standard convolution is shown in Eq. (5):

$$r_c = \frac{C_{dsc}}{C_{std}} = \frac{1}{C_{out}} + \frac{1}{K^2} \tag{5}$$

The parameters cost ratio r_p of the Depthwise separable convolution to standard convolution is shown in Eq. (6).

$$r_p = \frac{P_{dsc}}{P_{std}} = \frac{1}{C_{out}} + \frac{1}{K^2} \tag{6}$$

Eqs. (5) and (6) show that the advantage of depth separable convolution is changed the multiplication operation into multiplication and addition operation, which can significantly reduce parameters and computational cost.

Inverted Residual Block added a 1 × 1 Pointwise convolution before Depthwise convolution, because Depthwise convolution, due to its computational properties, cannot change the number of channels on its own, so the number of output channels is the same as the number of input channels. The Pointwise convolutional layer can increase the dimension so that the network can extract features in a higher dimensional space, which can effectively improve the network performance. The input channel of the module is C_{input}, and the expansion factor n, the number of output channels after the point convolution layer is $n \times C_{input}$, it means the expansion multiple of the channel. Through the second Pointwise convolution layer, the dimension can be reduced so that the number of channels in the output is the same as that in the input. After dimension reduction, feature information can be concentrated in the channel after dimension reduction. If the nonlinear activation function ReLU is used, there will be a large loss of information, so the linear activation function is connected after dimension reduction. In the meantime, the module also contains a shortcut. Fig. 1 shows an Inverted Residual Block.

3.3. Network architecture

In this section, we will systematically describe the proposed method and introduce the techniques used in the various parts of the method. First, the method uses the properly sized enlarged image as input to the network. Second, the last some modules are replaced by an Inverted Residual Block with fewer parameters. Finally, the convolution layer, Batch Normalization layer, ReLU activation function, Global Average Pooling, and Softmax were added to form our network. Fig. 2 shows an improved network structure for improved CNN.

In this method, image classification mainly based on Softmax. Softmax is used to calculate the probability values of each category, and shown as:

$$p_i = Softmax = \frac{e^{z_i}}{\sum_j^J e^{z_j}} \tag{7}$$

where z_i the input of softmax, and j is the number of categories. With softmax function, the output value of multi-classification can be transformed into probability distribution with the range of (0,1) and sum of 1. The cross entropy loss function of Softmax can be used to determine the proximity between actual and desired output, the formula is as follows:

$$L = \sum_j^J y_j \log p_i \tag{8}$$

where, y_j represents the real tag. For the classification problem, the loss function reflects the difference between the predicted result and the actual result. The training process of the network can be regarded as a parameter optimization process, that is to find a group of optimal solutions in the parameter space to make L minimize.

4. Experiment results and analysis

4.1. Datasets

In this paper, five widely used classification benchmarks are adopted for evaluation: 2 fine-grained biological image classification datasets (Caltech-UCSD Birds 2011–200 (Wah et al., 2010), 102 Category Flower (Nilsback and Zisserman, 2008)). 2 general classification datasets (Caltech-101 (Fei-Fei et al., 2004), Caltech-256 (Griffin et al., 2007)), these two datasets also contain biological images. 1 Scene recognition dataset (Indoor-67 (Quattoni and Torralba, 2009)). In order to increase the input image size, and prevent too large image size will lead to a sharp increase in computation, we first calculated the

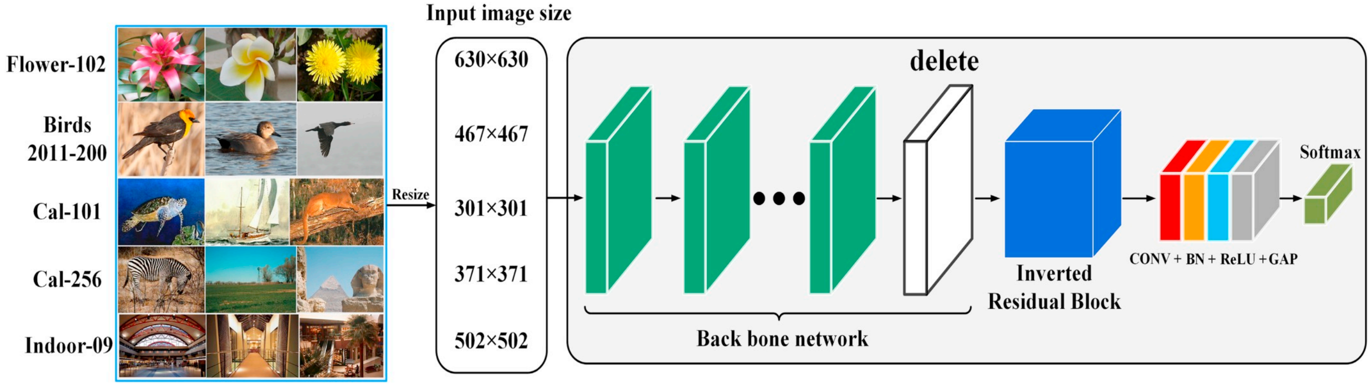


Fig. 2. The network architecture of improved CNN.

maximum values of the average length and width of images in each dataset, which are shown in Table 1, and then select the maximum value as the size of the input image. For example, for the Flower-102 dataset, the maximum value of the average length and width is 630, so we set the input image size of the network to (630,630).

For general classification datasets Caltech-101 and Caltech-256, we randomly selected 30 and 60 images from each category as the training set, respectively, and the rest as the validation set. For the Scene recognition dataset Indoor-67, we randomly select 80 pictures from each category as the training set, and the rest as the validation set. For the fine-grained image classification datasets Flower-102 and Caltech-UCSD birds-200-2011 datasets, we randomly selected 20 and 30 images from each category as training sets, respectively, and the rest as the validation set. To prevent the random partitioning of training sets and validation sets from influencing the results, we randomly selected the training/test images for three times and reported the average classification accuracy. For the Birds-200-2011 dataset, we did not use the bounding boxes and part annotations given by the dataset. Sample images of different datasets are shown in Fig. 3.

4.2. Training

In this paper, the Keras framework is adopted. The experiments were conducted with the Intel(R) Core (TM) i7-7800X CPU @ 3.50GHz, 64.00 GB RAM and one Nvidia GeForce GTX 1080 Ti GPU.

The Inverted Residual Block contained shortcut structure, but in MobileNetV2, not all the blocks were used, When the *stride* = 1 the block did not contain the shortcut structure, while when the *stride* = 2 did the opposite. But we just replaced the last Inception module, so we used the Inverted Residual Block without the shortcut structure.

We selected two networks as the back bone network in the experiment. For the baseline is InceptionV3, we replaced last Inception module, width multiplier α set as 1, and its main work is to refine the network evenly at each layer. The hyperparameters expansion factor n in Inverted Residual Block was to controls the multiple of dimension expansion. In our experiment, the expansion factor n are set as 1,2,3 for comparison experiment. The number of filters in the Inverted Residual Block is 512. The batch size is set to 8 and epochs are set to 30.

The main structure of the DenseNet201 consists of four different Dense blocks and the transition Layer (Huang et al., 2017) which connect each Dense block. Each Dense Block consists of 6, 12, 48, and 32 Bottleneck layers, respectively. Each Bottleneck layer consists of BN-ReLU-Conv(1 × 1)-BN-ReLU-Conv(1 × 1), where BN is the Batch Normalization, ReLU is the rectified linear unit and Conv(1 × 1) is the convolution of 1 × 1. For the baseline is DenseNet201, in the last Dense Block, we removed 14 Bottleneck layers with one Inverted Residual Block. For Cal-101 and Cal-256, the batch size is set to 8, for Flower-102, the batch size is set to 2, and for another dataset, the batch size is set to 4. The other parameters are set the same as a baseline were InceptionV3.

Table 1
Average side length and size of input image on different datasets.

Datasets	Length of image	Width of image	Input image size
Cal-101	301	244	301 × 301
Cal-256	371	325	371 × 371
Flower-102	630	534	630 × 630
Birds-200-2011	467	386	467 × 467
Indoor-67	502	417	502 × 502

In fine-tuning, Training set and validation set are randomly selected from dataset. The network is trained using stochastic gradient descent (SGD), and momentum is set to 0.9. The batch size is set to 8 and epochs are set to 30. The initial learning rate is set to 0.001 and multiplied by 0.94 every two epochs. Our initial weight adopts the weight of InceptionV3 on the ImageNet.

4.3. The effect of input image size

We first evaluated the impact of image size on the classification performance of the InceptionV3 without any modification to the network structure. To increase the input image size, and prevent too large image size will lead to a sharp increase in computation, we choose the enlarged input image size in Table 2 as the network input, and compared classification accuracy with the original InceptionV3 input size (299 × 299). The experimental results are shown in Table 2.

As can be seen from Table 2, increasing the size of the image can improve the classification accuracy. This is because expanding the size of the input image can effectively save the image features and prevent information loss. For the general object classification datasets Cal-101 and Cal-256, the accuracy is less improved, Because of the enlarged image size close to the original image size. For fine-grained biological image datasets Flower-102 and Caltech-UCSD Birds 2011-200, as well as scene classification dataset Indoor-67, the accuracy improves significantly (+2.06%, +4.71%, +5.65%). This is because the differences between the categories of these datasets are more subtle. It is often only using small local differences that different categories can be distinguished. While using the original network input size, there is a high probability that this local information of images will be lost. By increasing the input image size, image information loss can be effectively prevented, local information of images loss and effective features of the image can effectively save the distortion can be effectively prevented, and classification accuracy can be improved.

4.4. The effect of expansion factor n

The expansion factor n controls the ascending dimension multiple in Inverted Residual Block, enabling the network to extract features in

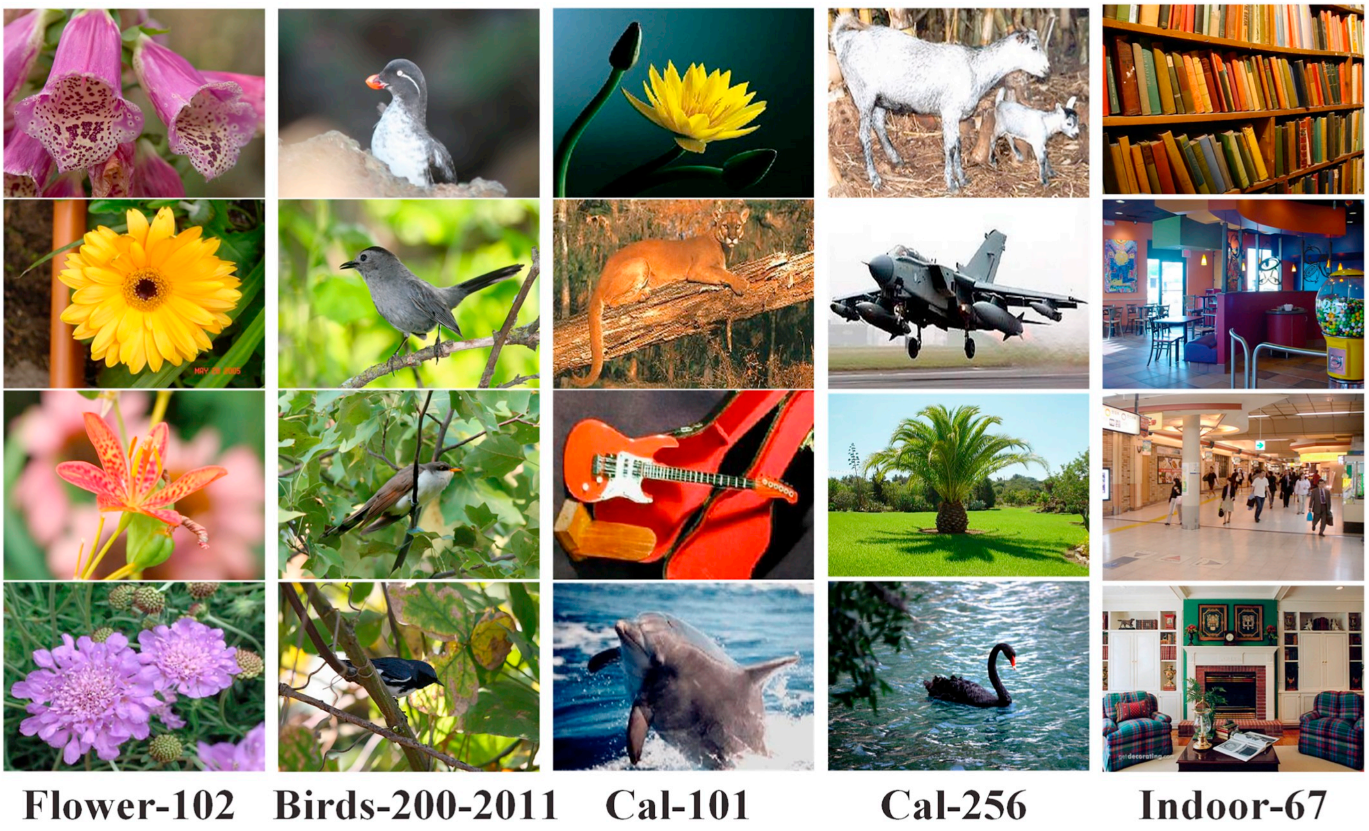


Fig. 3. Sample images from the different experimental datasets.

Table 2
Comparison of classification accuracy(%) of different input image size.

Datasets	Accuracy	
	(Original input image size)	(Enlarged input image size)
Cal-101	94.31(299 × 299)	94.31(301 × 301)
Cal-256	85.59(299 × 299)	86.65(371 × 371)
Flower-102	93.80(299 × 299)	95.86(630 × 630)
Birds-200-2011	76.96(299 × 299)	81.67(467 × 467)
Indoor-67	70.90(299 × 299)	76.55(502 × 502)

higher dimensions. Meanwhile, the influence of expansion factor $n=1$, $n=2$, and $n=3$ on experimental results is compared in baseline as InceptionV3. The implementation results are shown in Table 3. However, for different datasets, the categories are different, and the parameters in the Softmax layer will be different, so the comparison experiment does not include the parameters of Softmax.

As can be seen from Table 3, in our method, the selection of 1, 2, and 3 of the hyperparameters expansion factor n in the Inverted Residual Block did not significantly affect the final classification accuracy. However, the parameter $n=2$ and $n=3$ are much higher than that of $n=1$, but the classification accuracy is similar. This may be because the features of the higher level are semantic features, and the extension of the dimension of the higher-level features has little impact on the experimental results. But in the deep layer of the network, if the n were too large, the network parameters would rise sharply. So we choose the parameter $n=1$, which not only has a better classification effect than the original network but also has fewer parameters.

4.5. Comparison of parameters and flops

In Table 4, we compare our method with the parameters and FLOPs (Floating point operations) of the original InceptionV3 and DenseNet. In this comparison experiment, we also do not include the parameters of Softmax.

As can be seen from Table 4, our method has fewer parameters than

Table 3
Comparison of accuracy (%) and parameters of different n on five datasets with baseline as InceptionV3.

Datasets	n = 1		n = 2		n = 3	
	Accuracy	Parameter	Accuracy	Parameter	Accuracy	Parameter
Cal-101	94.19	**21,530,848**	**94.46**	26,808,544	94.46	32,086,240
Cal-256	86.62		86.20		**86.86**	
Flower-102	96.58		96.24		**96.81**	
Birds-200-2011	81.88		82.27		**82.62**	
Indoor-67	77.33		77.50		**77.81**	

Table 4
Comparison of parameters of different methods.

Methods	Parameter count
Our method(baseline = InceptionV3)	21,530,848
Our method(baseline = DenseNet201)	18,162,304
InceptionV3	21,802,784
DenseNet201	18,321,984

Table 5
Comparison of FLOPs of different methods on five datasets.

Datasets	Our method	Our method	InceptionV3	DenseNet201
	Baseline: Inception	Baseline: DenseNet	(Simonyan and Zisserman, 2016)	(Huang et al., 2017)
Cal-101	194.60	**163.76**	198.00	165.86
Cal-256	196.03	**165.19**	200.86	168.54
Flower-102	194.60	**163.76**	198.00	165.86
Birds-200-2011	195.50	**164.66**	199.81	167.56
Indoor-67	194.27	**163.44**	197.36	165.26

the original network. This is because we use the module with fewer parameters to replace original module, which makes the parameters lower.

FLOPs(floating-point operations) are often used to measure algorithm or model complexity. We use the calculation method of FLOPs in TensorFlow. Since the number of categories for each dataset are different, so the parameters of Softmax are different, which makes the FLOPs of different datasets with the different number of categories are different. It is clear that our method is superior to corresponding original InceptionV3 and DenseNet201 for the complexities in Table 5.

4.6. *Comparison with the state-of-the-arts*

We compare our results to state-of-the-arts in image classification. To demonstrate the generality of our method, we chose the baseline networks as InceptionV3 and DenseNet201. We first compared our results with the advanced method and also compared with the most advanced network, such as DenseNet201. The experimental results are shown in Table 6.

The classification accuracy with a different method on five benchmark datasets is reported in Table 6, where we can see that our method of different baseline obtains the improved to different degrees by comparison to the method (Zheng et al., 2016), and our method with baseline as DenseNet generally obtains the higher results in most case. Compared with InceptionV3 and DenseNet201, the classification accuracy of all datasets has been improved to different degrees except Cal-101. This is because the size of the enlarged image on the Cal-101 dataset is similar to the original image, so our method has no advantage in classification accuracy.

In order to deal with the possible problems caused by randomly divided training sets and validation sets, we randomly divided each dataset three times. Fig. 4 shows the training curve of our improved network trained on five randomly divided datasets for the first time on different methods. The results of the different datasets are reported on the validation set with 30 epochs.

Table 6
Comparison of classification accuracy(%) with the State-of-the-arts.

Datasets	Our method (baseline: Inception)	Our method (baseline: DenseNet)	Conv5 Feature (Zheng et al., 2016)	InceptionV3 (Simonyan and Zisserman, 2016)	DenseNet201 (Huang et al., 2017)
Cal-101	94.18	**94.31**	92.30	**94.31**	94.24
Cal-256	**86.62**	86.24	86.00	85.59	86.49
Flower-102	96.58	**97.32**	95.60	93.80	95.19
Birds-200-2011	81.88	**83.93**	76.40	76.96	80.57
Indoor-67	77.33	**78.76**	78.40	70.90	73.40

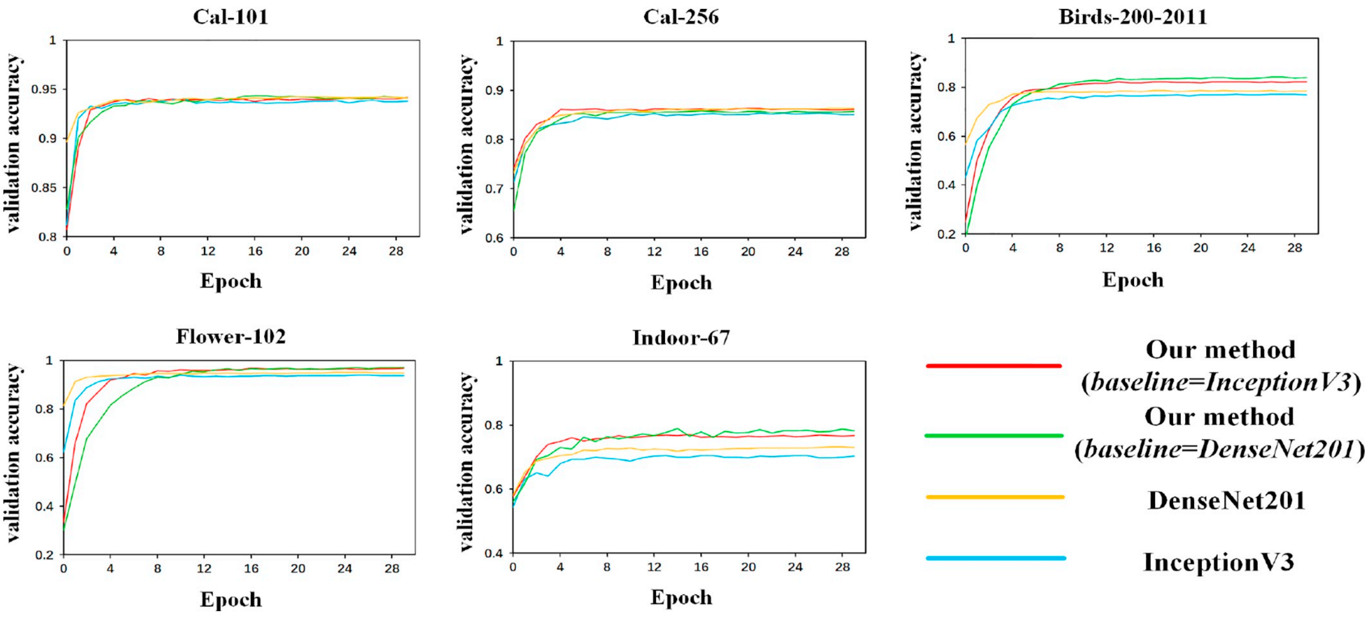


Fig. 4. Training profile of different methods on five datasets.

As can be seen from Fig. 4, where we can clearly see that our method generally obtains the best results in most cases. The accuracy of our method in dataset Flower-102, Birds-200-2011 and Indoor-67 tend to be higher than that of DenseNet201, InceptionV3. However, the classification accuracy in the datasets Cal-101 and Cal-256 is similar to that of DenseNet201 and InceptionV3.

In a word, the experiment shows the superiority of our method, which can not only effectively improve the classification accuracy, but also reduce the parameters and FLOPs.

5. Conclusion

In this paper, we studied a challenging in biological image classification, and a biological image classification method was proposed to handle this task. To improve classification accuracy, the size of the image input was increased. To solve the problem of increased computation caused by this method, the some modules in CNN was replace by Inverted Residual Block. The method was validated on five benchmark datasets, two of which are biological image datasets, and it yielded the promising accuracy, and less parameters. The experiment also proves method scalability, which can be applied to different networks. In future work, we hope to find a more efficient and lightweight convolution method that can more effectively reduce network parameters.

Funding

This work was supported in part by the National Natural Science Foundation of China under Grant 61772561, in part by the Key Research and Development Plan of Hunan Province under Grant 2018NK2012 and 2019SK2022, in part by the Science Research Projects of Hunan Provincial Education Department under Grant 18A174 and 18C0262, in part by the Degree & Postgraduate Education Reform Project of Hunan Province under Grant 2019JGYB154; in part by the Postgraduate Excellent Teaching Team Project of Hunan Province under Grant [2019]370-133 and in part by the Postgraduate Education and Teaching Reform Project of Central South University of Forestry & Technology under Grant 2019JG013.

References

Balfoort, H.W., Snoek, J., Smiths, J.R.M., Breedveld, L.W., Hofstraat, J.W., Ringelberg, J., Jul 1992. Automatic identification of algae: neural network analysis of flow cytometric data. J. Plankton Res. 14 (4), 575–589.

Chan, T.H., Jia, K., Gao, S., Dec 2015. PCANet: a simple deep learning baseline for image classification? IEEE Trans. Image Process. 24 (12), 5017–5032.

Chollet, F., 2017. Xception: deep learning with depthwise separable convolutions. In: Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), pp. 1800–1807.

Fei-Fei, L., Fergus, R., Perona, P., 2004. Learning generative visual models from few training examples: an incremental Bayesian approach tested on 101 object categories. CVPR 2004. In: Workshop on Generative-Model Based Vision. IEEE.

Griffin, G., Holub, A., Perona, P., 2007. Caltech-256 Object Category Dataset. Technical report. .

Hinton, G., Vinyals, O., Dean, J., 2015. Distilling the knowledge in a neural network. In: In: arXiv preprint arXiv:1503.02531.

Howard, A.G, Zhu, M.L, Chen, B., Kalenichenko, D., Wang, W.J., Wyand, T., Anderrtto, M., Adam, H., 2017. Mobilenets: Efficient convolutional neural networks for mobile vision applications. In: In: arXiv preprint arXiv:1704.04861.

He, K.M., Zhang, X.Y., Ren, S.Q., Sun, J., 2016. Deep residual learning for image recognition. In: Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), pp. 770–778.

He, K.M., Zhang, X.Y., Ren, S.Q., Sun, J., Sep 2015. Spatial pyramid pooling in deep convolutional networks for visual recognition. IEEE Trans. Pattern Anal. Mach. Intell. 37 (9), 1904–1916.

Huang, G., Liu, Z., van der Maaten, L., Weinberger, K.Q., Jul 2017. Densely connected convolutional networks. In: Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), pp. 2261–2269.

Jiao, L.C., Liu, F., Jul 2016. Wishart deep stacking network for fast POLSAR image classification. IEEE Trans. Image Process. 25 (7), 3273–3286.

Kingma, D.P., Ba, J., 2014. Adam: A method for stochastic optimization. In: In: arXiv preprint arXiv:1412.6980.

Krizhevsky, A., Sutskever, I., Hinton, G.E., 2012a. Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, pp. 1097–1105.

Krizhevsky, A., Sutskever, I., Hinton, G.E., 2012. Very deep convolutional networks for large-scale image recognition. In: In: arXiv preprint arXiv:1409.1556.

Kumar, A., Kim, J., Lyndon, D., Fulham, M., Feng, D.G., 2016. An ensemble of fine-tuned convolutional neural networks for medical image classification. IEEE J. Biomed. Health Inf. 21 (1), 31–40.

Lin, M., Chen, Q., Yan, S.C, 2013. Network in network. In: In: arXiv preprint arXiv:1312.4400.

Liu, Q., Xiang, X.Y., Qin, J.H., Tan, Y., Luo, Y.J., 2020. Coverless steganography based on image retrieval of DenseNet features and DWT sequence mapping. Knowl.-Based Syst. 192, 105375–105389. https://doi.org/10.1016/j.knosys.2019.105375.

Long, J., Shelhamer, E., Darrell, T., Jun 2015. Fully convolutional networks for semantic segmentation. In: Proc. IEEE Conf. Comput. Vis. Pattern Recognit. CVPR, pp. 3431–3440.

Luo, Y.J., Qin, J.H., Xiang, X.Y., Tan, Y., Liu, Q., Xiang, L.Y., 2019. Coverless realtime image information hiding based on image block matching and Dense Convolutional Network. J. Real-Time Image Proc. 17 (1), 125–135. https://doi.org/10.1007/s11554-019-00917-3.

Ma, W.T., Qin, J.H., Xiang, X.Y., Tan, Y., Luo, Y.J., Xiong, N.N., 2019. Adaptive median filtering algorithm based on divide and conquer and its applicationin CAPTCHA recognition. CMC-Comput. Mater. Continua 58 (3), 665–677. https://doi.org/10.32604/cmc.2019.05683.

Mao, Z.H., Han, S., Pool, J., Li, W.S, Liu, X.Y., Wang, Y., Dally, W.J, 2017. Exploring the regularity of sparse structure in convolutional neural networks. In: In: arXiv preprint arXiv:1705.08922.

Mohammad, R., Vicente, O., Joseph, R., Ali, F., Aug 2016. XNOR-net: ImageNet classification using binary convolutional neural networks. In: Computer Science, pp. 525–542.

Nilsback, M.-E., Zisserman, A., 2008. Automated flower classification over a large number of classes. In: Proceedings of the Indian Conference on Computer Vision, Graphics and Image Processing.

Noh, H., Hong, S., Han, B., 2015. Learning deconvolution network for semantic segmentation. In: IEEE International Conference on Computer Vision (ICCV). Dec 2015. pp. 1520–1528.

Pan, W.Y, Qin, J.H, Xiang, X.Y, Wu, Y., Tan, Y., Xiang, L.Y., 2019. A Smart Mobile Diagnosis System for Citrus Diseases Based on Densely Connected Convolutional Networks. IEEE Access 7 (1), 87534–87542. https://doi.org/10.1109/ACCESS.2019.2924973.

Pan, S.J., Yang, Q., Oct 2010. A survey on transfer learning. In: IEEE Transactions on Knowledge and Data Engineering. 22(10). pp. 1345–1359.

Pan, L.L., Qin, J.H., Chen, H., Xiang, X.Y., Li, C., Chen, R., 2019. Image augmentation-based food recognition with convolutional neural networks. CMC-Comput. Mater. Continua 59 (1), 297–313. https://doi.org/10.32604/cmc.2019.04097.

Quattoni, A., Torralba, A., 2009. Recognizing indoor scenes. In: Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), pp. 413–420.

Ruder, S., 2016. An overview of gradient descent optimization algorithms. In: In: arXiv preprint arXiv:1609.04747.

Sandler, M., Howard, A., Zhu, M.L., Zhmoginov, A., Chen, L., 2018. MobileNetV2: inverted residuals and linear bottlenecks. In: Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), pp. 4510–4520.

Sangineto, E., Nabi, M., Culibrkand, D., Sebe, N., Mar 2019. Self paced deep learning for weakly supervised object detection. IEEE Trans. Pattern Anal.Mach. Intell. 41 (3), 712–725.

Schneider, S., Taylor, G.W., Kremer, S., Jul 2018. Deep learning object detection methods for ecological camera trap data. In: 2018 15th Conference on Computer and Robot Vision (CRV). IEEE, pp. 321–328.

Simonyan, K., Zisserman, A., 2016. Rethinking the inception architecture for computer vision. In: Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), pp. 2818–2826.

Simpson, R., Culverhouse, P.F., Ellis, R., Williams, R., 1991. Classification of Euceratium Gran. in neural networks. In: IEEE International Conference on Neural Networks in Ocean Engineering, pp. 223–230.

Sun, G.J., Yang, B., Yang, Z.Q., Xu, G.N., 2018. An adaptive differential evolution with combined strategy for global numerical optimization. In: Soft Computing, pp. 1–20.

Tajbakhsh, N., Shin, J.Y., Gurudu, S.R., Hurst, R.T., Kendall, C.B., Gotway, M.B., Liang, J.M., May 2016. Convolutional neural networks for medical image analysis: full training or fine tuning? IEEE Trans. Med. Imaging 35 (5), 1299–1312.

Tian, B., Li, L., Qu, Y.S., Yan, L., Aug 2017. Video object detection for tractability with deep learning method. In: 2017 Fifth International Conference on Advanced Cloud and Big Data (CBD), pp. 36–45.

Wah, C., Branson, S., Welinder, P., Perona, P., Belongie, S., 2010. Caltech-UCSD Birds 200. CVPR Workshop on FineGrained Visual Categorization.

Wang, G., Li, W.Q., Zuluaga, M.A., Pratt, R., Patel, P.A., Aertsen, M., Doel, T., David, A.L., Deprest, J., Ourselin, S., Vercauteren, T., Jul 2018. Interactive medical image segmentation using deep learning with image-specific fine tuning. In: IEEE Transactions on Medical Imaging. 37(7). pp. 1562–1573.

Wang, J., Qin, J.H., Xiang, X.Y., Tan, Y., Pan, N., Jun 2019. CAPTCHA recognition based on deep convolutional neural network. Math. Biosci. Eng. 16 (5), 5851–5861. https://doi.org/10.3934/mbe.2019292.

Zhang, X., Zhou, X., Lin, M., Sun, J., 2018. ShuffleNet: an extremely efficient convolutional neural network for mobile devices. In: Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), pp. 6848–6856.

Yuan, Y., Mou, L.C., Lu, X.Q., 2015. Scene Recognition by Manifold Regularized Deep Learning Architecture. IEEE Trans. Neural Networks and Learning Systems 26 (10), 2222–2233.

Yuan, C.S., Xia, Z.H., Sun, X.M., Wu, Q.M.J., 2019. Deep Residual Network with Adaptive Learning Framework for Fingerprint Liveness Detection. IEEE Trans. Cognitive and Developmental Systems. https://doi.org/10.1109/TCDS.2019.2920364.

Zeiler, M.D., Fergus, R., 2014. Visualizing and understanding convolutional networks. In: European conference on computer vision. Springer, pp. 818–833.

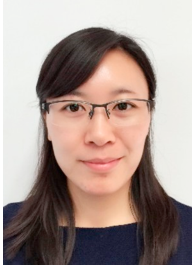
Zhang, J.M., Wang, W., Lu, C.Q., Wang, J., Sangaiah, A.K., 2019. Lightweight deep network for traffic sign classification. In: In: Annals of Telecommunications online.

Zheng, L., Zhao, Y.L., Wang, S.J., Tian, Q., 2016. Good practice in CNN feature transfer. In: In: arXiv preprint arXiv:1604.00133.

Jiaohua Qin received her B.S. degree in mathematics from the Hunan University of Science and Technology, China, in 1996, the M.S. degree in computer science and technology from the National University of Defense Technology, China, in 2001, and the Ph.D. degree in computing science from Hunan University, China, in 2009. She was a Visiting Professor with the University of Alabama, Tuscaloosa, AL, USA, from 2016 to 2017. She is currently a Professor with the College of Computer Science and Information Technology, Central South University of Forestry and Technology, China. Her research interests include network and information security, machine learning and image processing.

Wenyan Pan received his BS in Electronic and Information Engineering from Hunan City University, China, in 2017. He is currently pursuing his MS in Computer Technology at College of Computer Science and Information Technology, Central South University of Forestry and Technology, China. His research interests include machine learning and image processing.

Xuyu Xiang received his B.S. in mathematics from Hunan Normal University, China, in 1996, M.S. degree in computer science and technology from National University of Defense Technology, China, in 2003, and PhD in computing science from Hunan University, China, in 2010. He is a professor with the College of Computer Science and Information Technology, Central South University of Forestry and Technology, China. His research interests include network and information security, image processing and machine.



Yun Tan received her M.S. and Ph.D. degrees both from Beijing University of Posts and Telecommunications, China, in 2004 and 2016, respectively. Now she is a lecturer with College of Computer Science and Information Technology, Central South University of Forestry and Technology, China. Her research interests mainly include image security, compressive sensing and signal processing.

Guimin Hou received her BS in Communication Engineering from Central South University of Forestry and Technology, College of Computer and Information Engineering, China, in 2019. She is currently pursuing her MS in information and communication engineering at College of Computer and Information Engineering, Central South University of Forestry and Technology, China. Her research interests include pattern recognition and image processing.