Automation, Control and Intelligent Systems
2021; 9(1): 46-60
http://www.sciencepublishinggroup.com/j/acis
doi: 10.11648/j.acis.20210901.16
ISSN: 2328-5583 (Print); ISSN: 2328-5591 (Online)

Research on Face Recognition Algorithm Based on Improved Residual Neural Network

Tang Xiaolin[1, 2], Wang Xiaogang[1, 2, *], Hou Jin[1, 2], Han Yiting[1, 2], Huang Ye[1, 2]

[1]School of Automation & Information Engineering, Sichuan University of Science & Engineering, Yibin, China

[2]Artificial Intelligence Key Laboratory of Sichuan Province, Sichuan University of Science & Engineering, Yibin, China

Email address:
wxg_zf@163.com (Wang Xiaogang)

[*]Corresponding author

To cite this article:
Tang Xiaolin, Wang Xiaogang, Hou Jin, Han Yiting, Huang Ye. Research on Face Recognition Algorithm Based on Improved Residual Neural Network. *Automation, Control and Intelligent Systems*. Vol. 9, No. 1, 2021, pp. 46-60. doi: 10.11648/j.acis.20210901.16

Received: March 18, 2021; **Accepted**: March 30, 2021; **Published**: April 12, 2021

Abstract: The residual neural network is prone to two problems when it is used in the process of face recognition: the first is "overfitting", and the other is the slow or non-convergence problem of the loss function of the network in the later stage of training. In this paper, in order to solve the problem of "overfitting", this paper increases the number of training samples by adding Gaussian noise and salt and pepper noise to the original image to achieve the purpose of enhancing the data, and then we added "dropout" to the network, which can improve the generalization ability of the network. In addition, we have improved the loss function and optimization algorithm of the network. After analyzing the three loss functions of Softmax, center, and triplet, we consider their advantages and disadvantages, and propose a joint loss function. Then, for the optimization algorithm that is widely used through the network at present, that is the Adam algorithm, although its convergence speed is relatively fast, but the convergence results are not necessarily satisfactory. According to the characteristics of the sample iteration of the convolutional neural network during the training process, in this paper, the memory factor and momentum ideas are introduced into the Adam optimization algorithm. This can increase the speed of network convergence and improve the effect of convergence. Finally, this paper conducted simulation experiments on the data-enhanced ORL face database and Yale face database, which proved the feasibility of the method proposed in this paper. Finally, this paper compares the time-consuming and power consumption of network training before and after the improvement on the CMU_PIE database, and comprehensively analyzes their performance.

Keywords: Residual Neural Network, Data Enhancement, Overfitting, Loss Function, Optimization Algorithm

1. Introduction

The groundbreaking research on neural networks can be traced back to the 1980s. In 1980, Fukushima of Kyoto University in Japan proposed a deep-structure neural network named "neocognitron" [1] based on the visual cortex. In 1980, Rumelhart proposed the error back propagation algorithm [2], which makes the training of neural networks possible. In 1989, Yann LeCun introduced error back propagation in the neural network for recognition of handwritten digits [3]. In 1998, he published his paper to show the world LeNet5 [4], and for the first time proposed the term "convolution", which led to the name "convolutional neural network". In the 2012 ImageNet Large-scale Visual Recognition Challenge, AlexNet designed by Hinton and his student Alex Krizhevsky won the championship [5, 6]. Compared with LeNet, the AlxNet network is wider and deeper. In the next few years, convolutional neural networks developed rapidly. In the 2014 ILSVRC competition, GoogleNet and VGGNet won the championship and runner-up respectively [7-9]. In the 2015 competition, ResNet with better performance and fewer parameters won the championship [10].

Convolutional Neural Network (CNN) is a type of feedforward neural network that includes convolution calculations, and it has a deep structure [11], which is a typical method of deep learning.

At present, more and more convolutional neural networks are widely used in the field of face recognition [12]. In 2014,

Facebook released Deepface based on convolutional neural networks, the recognition rate of this network in the LFW face data set is as high as 97.35%. [13]. The recognition accuracy of DeepID [14] proposed by Yi Sun of the Chinese University of Hong Kong on the LFW dataset reached 97.45%. In 2015, Google's FaceNet [15] was better than DeepID, and the accuracy of face recognition on the same face data set reached 99.63%. However, although ResNet cannot achieve such a high recognition rate in face recognition, its network has fewer parameters and faster training speed, which makes it also has a great advantage in the field of face recognition. But its disadvantage is that when the number of samples in the face database is small, the network is prone to "overfitting", and the convergence of the loss function in the network training process is not good.

The main work of this paper is divided into the following four aspects:

1) Suppress the "overfitting" of the network by enhancing the face data set and introducing "dropout" in the residual network;

2) Solve the problem of poor convergence of the loss function during the training process of the network by proposing a joint loss function and improving the Adam optimization algorithm;

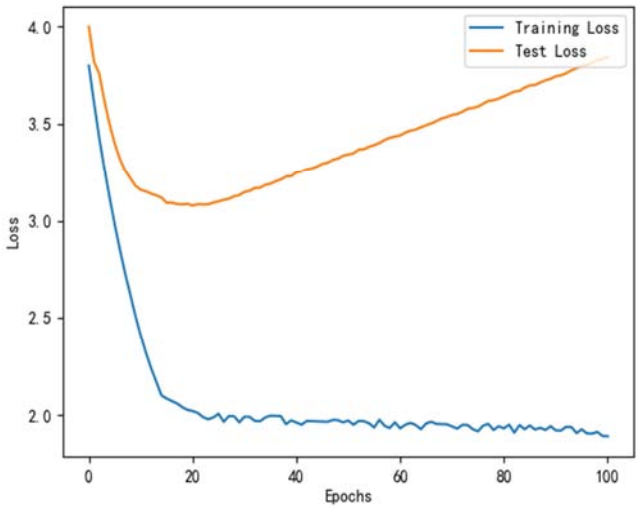
3) This paper has conducted verification experiments on the ORL face database and Yale B face database;

4) This paper compares the time-consuming and power consumption of the network on the CMU_PIE database with more data samples, so as to test the performance of the residual network before and after the improvement.

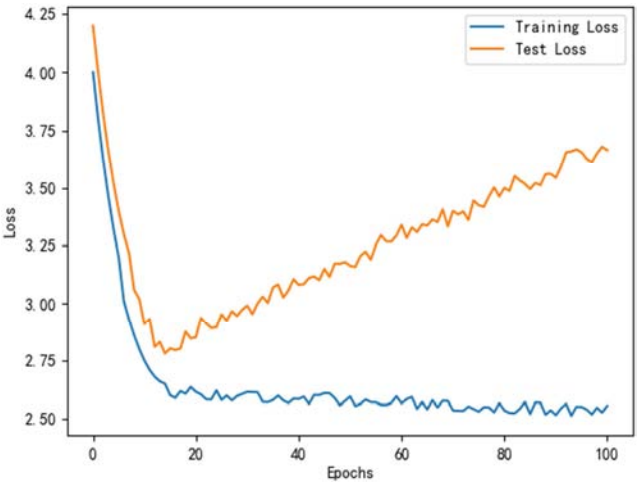
2. The "Overfitting" of the Residual Network

When the residual network is applied to face recognition, although it can achieve a relatively high recognition accuracy, on some face data sets, the network will have "overfitting" problems during training [16]. "Over-fitting" refers to during the network training process, because the model has too many parameters, and the numbers of sample is too small, which leads to too accurate learning and makes its recognition error rate on the training set is low or able to converge, but the error rate on the test set is high, or even does not converge. The "over-fitting" problem of the network will cause the accuracy of face recognition to be very unstable. Therefore, it is necessary for us to improve it.

There are many typical ResNet models in the residual network. Since it is not that the deeper the number of layers of the network, the better the effect. Therefore, considering the hardware conditions and actual requirements, compared to other networks, ResNet 18 and ResNet 34 has been more widely used, but when we use ResNet 18 in the residual network to perform face recognition on the ORL face database and the Yale face database, we set the batch-size to 20 during training. Then we initialize the parameters, after 100 epochs, the final loss curve is shown in Figure 1.



(a) Training results on ORL database



(b) Training results on Yale database

Figure 1. *Training results of ResNet 18 on the face database.*

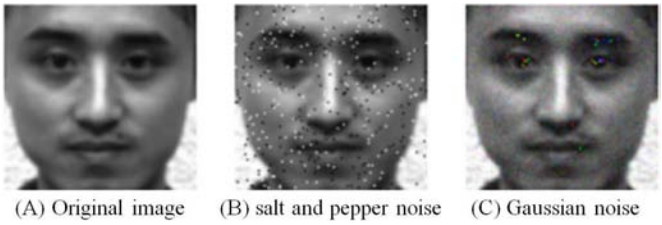
It can be seen from Figure 1 that although ResNet 18 has a lower loss on the training set on the two databases, its loss on the test set gradually increases with the increase in the number of training, which will lead to higher recognition error rate. This article prevents this "overfitting" of the network from two aspects. First of all, we enhance the face data, we artificially add Gaussian noise and salt and pepper noise to the image based on the original face image, so as to achieve the purpose of enriching the number and types of samples. Second, we introduce “dropout” into the network to improve the generalization ability of the network.

3. Prevention of Network "Overfitting"

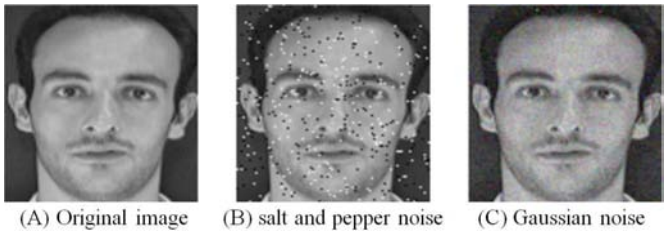
3.1. Enhance Face Data

An important method to improve the generalization ability of the network is to enhance the face data set [17], that is, on the basis of the original face image, we can rotate, stretch, adjust the contrast and color of the original image, and artificially add noise to get more images. In this paper,

Gaussian noise and salt and pepper noise are added to each image to increase the diversity of the samples in the face data set. The effect of adding noise to two of the samples is shown in Figure 2.



(a) Noisy image of samples in Yale database



(b) Noisy image of samples in ORL database

Figure 2. *The effect of adding noise to the training sample.*

The face data set enhancement can prevent the network from "overfitting" by increasing the breadth of the data. In addition, this paper adds “dropout” to the network to improve the generalization ability of the network by inactivating some neurons in the network.

3.2. Dropout

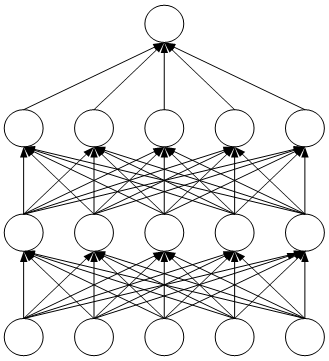
The principle of Dropout [18] can be explained as: in each training batch of the network, it weakens the interaction between the detectors by ignoring half of the feature detectors (making half of the hidden layer nodes 0).

The reasons for dropout to prevent network "overfitting" are mainly manifested in the following two aspects:

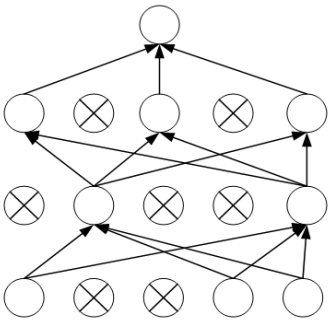
(1) Take the average, in the process of each iteration, dropout randomly inactivates a part of the neurons in the network, which is equivalent to the structure of the neural network is different when each batch is trained. This makes the network to get multiple different judgment results when judging the same sample, and the network can determine the category with the most judgment results as the correct classification.

(2) Reduce the complex co-adaptation relationship between neurons, because the introduction of dropout leads to two neurons not necessarily appearing in the same dropout network. In this way, the update of weights no longer depends on the joint action of implicit nodes with fixed relationships, which prevents some features from being effective only under other specific conditions. It forces the network to learn more robust features, which also exist in random subsets of other neurons.

The relationship between the neurons before and after adding dropout is shown in Figure 3.



(a) Network before adding dropout



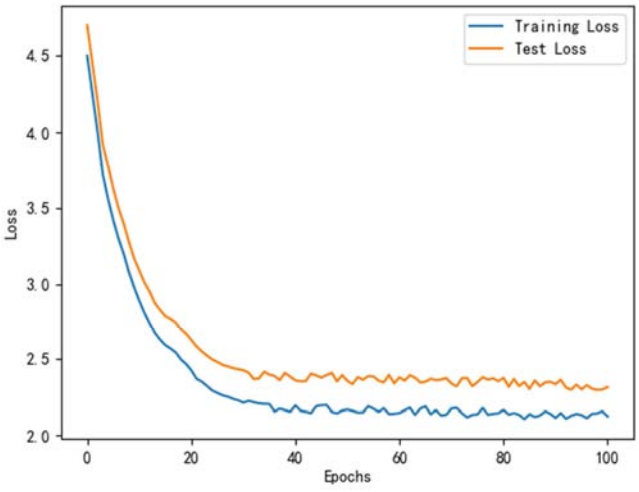
(b) Network after adding dropout

Figure 3. *Principle of dropout.*

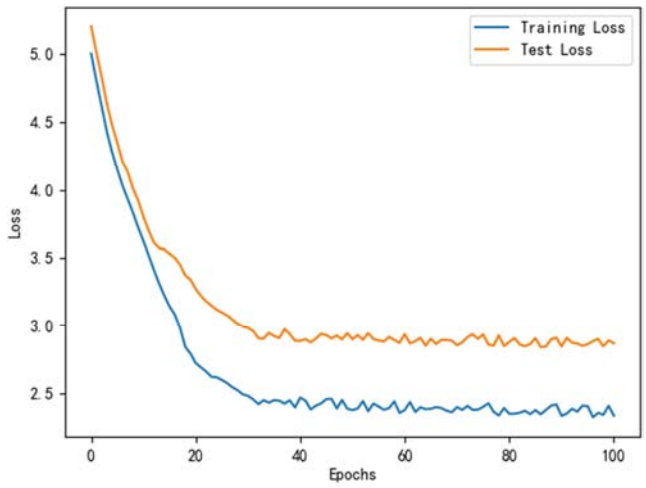
3.3. Verification to Prevent "Overfitting"

After the face data is enhanced and dropout is added to the network, this paper also conducts training and testing on ORL database and Yale database, and the results are shown in Figure 4.

It can be seen from Figure 4 that although this paper has solved the "overfitting" problem, because the number of samples in the database has greatly increased, the network needs more iterations to converge, that is, the loss function converges more slowly, therefore, this paper improves the loss function and optimization algorithm of the network to solve this problem.



(a) Training and testing results on ORL database



(b) Training and testing results on Yale database

Figure 4. *The results of training and testing of the improved network on the database.*

4. Loss Function

4.1. Training of Neural Network

This paper introduces the loss function through the understanding of the training process of the convolutional neural network. The convolutional neural network includes two parts: forward propagation and back propagation. During forward propagation, we select a training sample x and its corresponding class label y, and input x into the network. After each convolution, pooling, and fully connected layer, the final Softmax classifier will output an n-dimensional vector $h_{w,b}(x)$, n is the number of categories divided by the Softmax classifier, which is the number of persons that can be recognized when training face images. For example, the Yale B face database contains the face images of 10 persons, so n takes 10. In the vector $h_{w,b}(x)$, each element represents the probability that the input sample x belongs to each category. In the back propagation, we choose a loss function to calculate the square of the error between $h_{w,b}(x)$ and y, and update the parameters according to the principle of minimizing the square of error until the loss function approaches 0 or converges, the training of the network parameters is completed.

It can be seen from the above that the loss function is a very important part of the deep learning algorithm based on the multi-layer neural network model. The selection of the loss function will affect the accuracy of model training and the effect of convergence. This paper will consider the advantages of Softmax, Center, and Triplet loss functions to propose a joint loss function.

4.2. Softmax Loss

The proposal of Softmax Loss [19] is based on the Softmax function. The Softmax function can be explained by Figure 5. In Figure 5, first, the function performs a weighted summation on the input data $z_1 \cdots z_i$, and then each input data is divided by the summation to achieve the purpose of normalizing the input, because this function can normalize the input data to the interval [0 1], so the result can be used to represent the probability of each input.

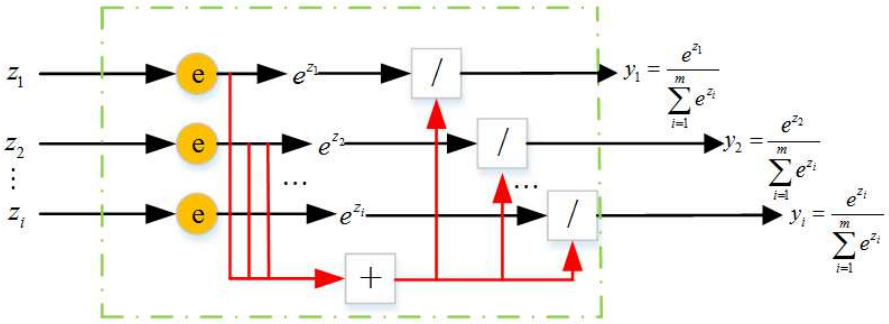


Figure 5. *The structure of the Softmax algorithm.*

The cross entropy loss of the Softmax function is defined as Softmax Loss, and the equation is as (1).

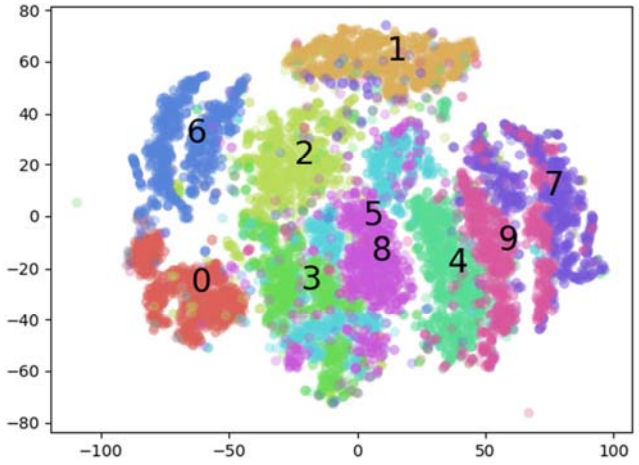
$$L_s = -\frac{1}{m}\sum_{i=1}^{m} log \frac{e^{W_{yi}^T x_i + b_{yi}}}{\sum_{i=1}^{m} e^{W_j^T x_i + b_j}} \quad (1)$$

Where, m is the batch-size in the training process, n is the total number of categories of samples in the database, W_j is the weight of the last layer of the network, x_i is the output of the penultimate layer of the network, and b is bias of the last layer.

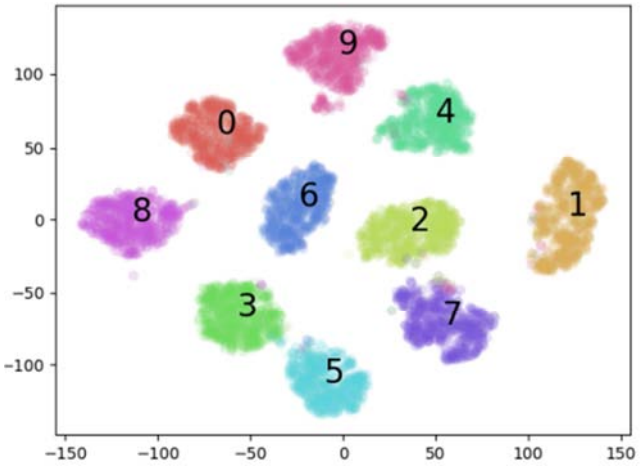
When the neural network classifies the samples, the output value of each Softmax represents the probability that the model predicts that the sample belongs to the class. If the value is larger, the probability that the sample belongs to the class is greater, the loss value on Softmax Loss is also smaller, otherwise the corresponding loss value is larger. Therefore, the model can increase the distance between different classes of samples according to the loss value, so as to classify the samples better.

In order to verify the classification effect of Softmax Loss, we conducted experiments on the MNIST data set. In Figure 6, (a) is the distribution map of the samples before classification, (b) is the effect map of Softmax Loss's classification of MNIST. From the comparison results in the figure, it can be seen that Softmax Loss classification can increase the distance between different samples, so that different samples can be distinguished. However, its main

disadvantage is that the intra-class distribution of the same sample is not compact enough. That is, the distance within the class is a bit large, so if the Softmax Loss is used alone in the network, the classification effect we get is not satisfactory.



(a) Distribution of samples before classification



(b) Classification results of Softmax Loss

Figure 6. *Comparison of results before and after classification.*

4.3. Center Loss

Center Loss [20] can ensure that the distance within a class is minimized under the premise that the features are separable, and the purpose is to improve the discriminability between features. Its main idea is to determine a center for each type of sample, and let the all samples in the class are close to this center, and samples that are far away must be punished. The equation is shown in (2),

$$L_c = \frac{1}{2m}\sum_{i=1}^{m} ||x_i - c_{yi}||_2^2 \quad (2)$$

In the equation, m is the batch-size in training, and c_{yi} refers to the center of the sample of yi. Equation (2) can be understood as: minimizing the sum of the squares of the distances between the elements in each class of samples and the center of the class, thereby reducing the distance of samples within the class.

Since Center Loss is mainly used to reduce the intra-class distance, it is not used alone. Generally, it is combined with Softmax Loss to increase the inter-class distance and reduce the intra-class distance. Their joint loss function is as follows as shown in equation (3).

$$\mathrm{L} = L_s + \lambda L_c = -\sum_{i=1}^{m} log\frac{e^{W_{yi}^T x_i + b_{yi}}}{\sum_{i=1}^{m} e^{W_j^T x_i + b_j}} + \frac{\lambda}{2}\sum_{i=1}^{m} ||x_i - c_{yi}||_2^2 \quad (3)$$

In equation (3), λ is used to control the weight of Center Loss and is a hyperparameter. In the experiment, we take λ as 0.1. The classification result of the joint loss function of Softmax Loss and Center Loss on MNIST is shown in Figure 7.

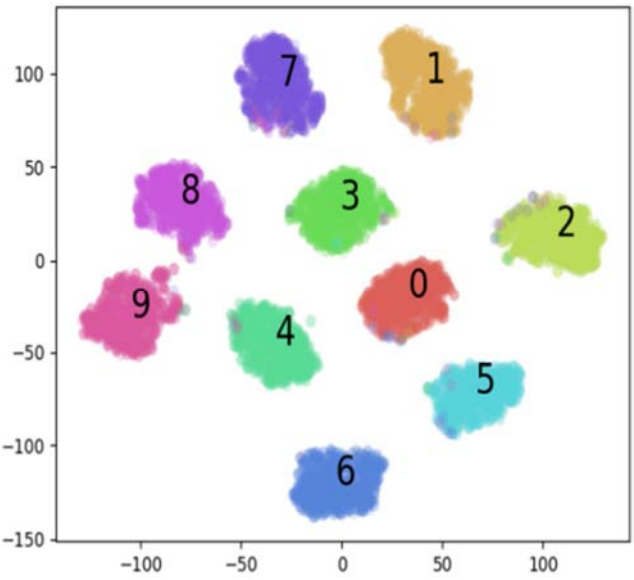


Figure 7. *Classification results of the joint loss function of Softmax Loss-Center Loss.*

It can be seen from Figure 7 that the classification effect of Softmax Loss-Center Loss on the MNIST dataset is better than that of Softmax Loss on the MNIST data set. This is reflected in the fact that the samples of the same class are more compactly distributed, and the distance within the class is greatly reduced and the distance between different types of samples is increased, so as to achieve the purpose of discrimination. However, the distance between certain classes is still very small, such as class 8 and class 9 in Figure 7. Therefore, the joint loss function of Softmax Loss-Center Loss needs to be further improved.

4.4. Triplet Loss

Triplet Loss [21] is also a commonly used loss function in neural networks, which can classify samples with small differences, such as face images. When it classifies the data set, the sample will be divided into three: positive sample, negative sample and anchor sample. Its main idea is to reduce the distance between the positive sample and the anchor sample, and increase the difference between the negative sample and the anchor sample. The optimization process is shown in Figure 8.

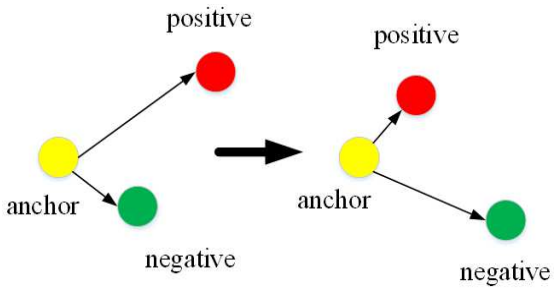


Figure 8. *The optimization process between each sample in Triplet Loss.*

Triplet Loss maps the input image to a feature space through f(x). This feature space is spherical. For an input image x^i, this function makes it as close as possible to its same type of image x^p and far away other types of images x^n. This can achieve the purpose of high cohesion and low coupling. The equation is shown in (4).

$$||f(x_j^i) - f(x_j^p)||_2^2 + \mu < ||f(x_j^i - f(x_j^n))||_2^2 \quad (4)$$

Among them, μ is margin, and equation (4) can be understood as the L2 distance between all negative samples and anchor samples in the sample image is greater than the distance between positive samples and anchor samples. The Triplet Loss function is shown in equation (5).

$$L_T = \sum_j^N [|f(x_j^i) - f(x_j^p)|_2^2 - |f(x_j^i) - f(x_j^n)|_2^2 + \mu]_+ \quad (5)$$

Where, "+" means that when the value in [] is greater than 0, the value is taken as the loss, and when it is less than 0, the loss is taken as 0. The classification result of Triplet Loss on the MNIST data set is shown in Figure 9.

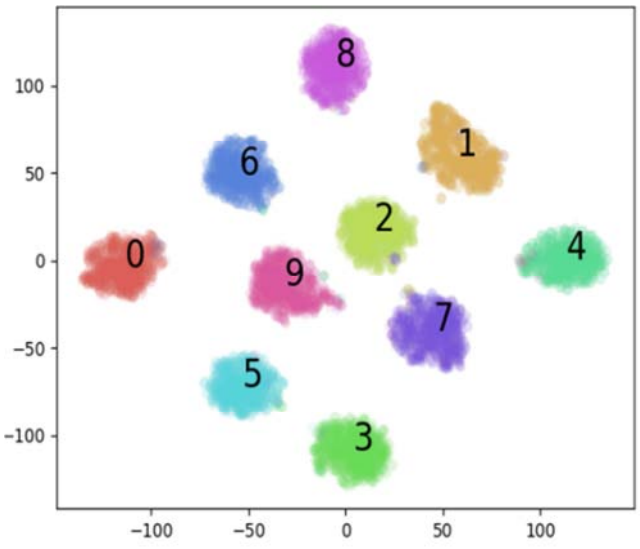


Figure 9. *Classification results of the Triplet Loss function.*

According to Figure 9, we can see that Triplet Loss has achieved better results in increasing the distance between classes of different samples. Therefore, it can compensate for the disadvantages of the joint loss function of Softmax Loss-Center Loss in calculating the distance between sample classes.

4.5. Joint loss Function

Comprehensive analysis of the advantages and disadvantages of the three loss functions of Softmax Loss, Center Loss, and Triplet Loss, this paper proposes a joint loss function. In the improved loss function, Center Loss is used to reduce the distance within the class, and Triplet Loss is used to increase the distance between large and inner areas, and Softmax Loss is used to ensure correct classification. The equation of the joint loss function is shown in (6).

$$L = L_s + \beta_1 L_c + \beta_2 L_T \quad (6)$$

Where, β_1 and β_2 are hyperparameters that control Softmax Loss, Center Loss, and Triplet Loss, respectively.

The specific steps of the joint loss function are: First, it gets the center position c_{yi} of each class through Center Loss, and finds the sample with the largest distance from c_{yi} in the class, and records it as c_{yimax}, and then calculates the difference between c_{yimax} and c_{yi}, the distance is recorded as S_1. Next, it selects the sample with the closest distance to c_{yi} as the anchor sample, and remembers this closest distance as S_{min}, After Triplet Loss optimizing the distance between samples, this function calculates the distance between the anchor sample and the nearest negative sample, the distance is recorded as S_2.

$$S_1 = ||c_{yimax} - c_{yi}||_2^2 \quad (7)$$

$$S_2 = ||f(x_\alpha) - f(x_n)||_2^2 \quad (8)$$

Among them, x_a is the selected anchor sample, and x_n is the negative sample closest to it. In order to get a better classification effect, we add the following judgments,

$$S_2 \geq 2S_1 - S_{min} \quad (9)$$

The schematic diagram of the above process is shown in Figure 10.

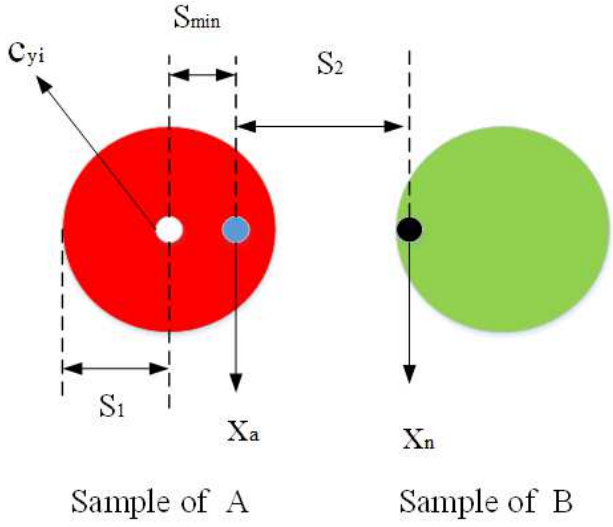


Figure 10. *Schematic diagram of the distance of the sample.*

If the judgment (9) is true, the loss value is the current loss

obtained, otherwise, the algorithm will add a penalty factor to the loss value.

In each batch of training of the neural network, Center Loss updates the center of each category through equations (10) and (11).

$$c_j^{t+1} = c_j^t - \gamma \Delta c_j^t \tag{10}$$

$$\Delta c_j^t = \frac{\sum_{i=1}^{m} \delta(y_i=j)(c_j - f_i)}{\sum_{i=1}^{m} \delta(y_i=j)+1} \tag{11}$$

Among them, t is the number of iterations, and γ is the class center learning rate. When the condition in parentheses after δ is satisfied, it takes 1; otherwise, it takes 0.

This paper proposes a joint loss function. The training process of the neural network is as follows: First, input the training sample $\{f_i\}$, set the number of classes of the sample to n, and initialize the center of each class of samples $\{c_j | j = 1,2 \cdots n-1, n\}$, hyperparameters β_1 and β_2, learning rate η_t, t is 1, at the beginning of training, initialize convolutional layer parameter θ_c^t and fully connected layer parameters W, the class center learning rate is γ. Specific steps are as follows:

1) Calculate the loss of the t-th iteration: $L^t = L_s^t + \beta_1 L_c^t + \beta_2 L_T^t$.;

2) Calculate the back propagation error of each sample: $\frac{\partial L^t}{\partial f_i^t} = \frac{\partial L_s^t}{\partial f_i^t} + \beta_1 \frac{\partial L_c^t}{\partial f_i^t} + \beta_2 \frac{\partial L_T^t}{\partial f_i^t}$;

3) Update parameter W: $W^{t+1} = W^t - \eta_t \frac{\partial L^t}{\partial W^t}$;

4) Update class center c_j: $c_j^{t+1} = c_j^t - \gamma \Delta c_j^t$;

5) Update the parameters of the convolutional layer θ_c: $\theta_c^{t+1} = \theta_c^t - \eta_t \sum_{i=1}^{N} \frac{\partial L^t}{\partial f_i^t} \frac{\partial f_i^t}{\partial \theta_c^t}$;

6) Number of iterations: $t = t + 1$;

The classification effect of the joint loss function on the MNIST data set is shown in Figure 11.

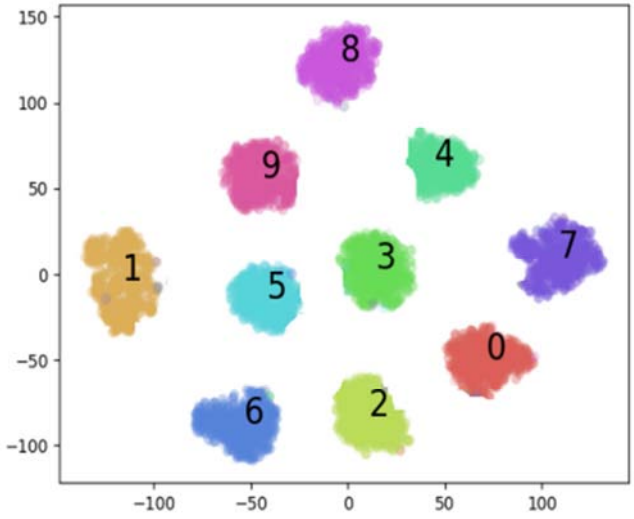


Figure 11. *The classification effect of the joint loss function.*

It can be seen from Figure 11 that the joint loss function proposed in this paper can simultaneously increase the distance between the classes of the sample while reducing the distance within the class of the sample when classifying the MNIST data set. The result of the classification is intuitive clearly, its discriminability is better.

After verifying the rationality of the joint loss function in sample classification, we also need to optimize the loss function. When the parameters are updated during the training process of the neural network, the process uses the gradient descent optimization algorithm. The selection of it will affect the convergence effect of the network and the accuracy of the face recognition. At present, the most widely used optimization algorithm is the Adam algorithm.

5. Optimization Algorithm

An optimization algorithm refers to an algorithm that minimizes the loss function by improving the training method of a neural network, which includes first-order optimization algorithm and second-order optimization algorithm. The first-order optimization algorithm updates the network parameters by calculating the first-order derivative and then according to the gradient descent method. In the same way, the second-order optimization algorithm updates the parameters of the network by calculating the second-order derivative. Compared with the second-order optimization algorithm, the first-order optimization algorithm has the obvious advantage of less calculation. Therefore, it has been more widely used in neural networks. At present, the most commonly used first-order optimization algorithm is the Adam algorithm.

5.1. Adam Algorithm

Adam optimization algorithm [22] is an extension of the stochastic gradient descent algorithm. In recent years, it has been widely used in deep learning, especially in the fields of computer vision and natural language processing. Adam algorithm is different from traditional stochastic gradient descent. Stochastic gradient descent updates all weights based on a single learning rate, and the learning rate does not change during the training process. Adam adjusts the learning rate adaptively by calculating the first-order moment estimation and the second-order moment estimation of the gradient.

The steps of Adam algorithm are as follows:

First, set the global learning rate η, the decay rate of the first and second moment estimation ξ_1 and ξ_2, a small constant δ. Then, set the time step t to 0, initialize the parameter θ, which includes the weight w and bias b, the first-order moment vector s_0 and the second-order moment vector r_0 are all initialized to 0. Then the algorithm selects m samples $\{(x^1, y^1), \cdots, (x^m, y^m)\}$ from the training set.

Calculate the gradient:

$$g_t = \frac{1}{m} \nabla_\theta \sum L(\theta, x^{(t)}, y^{(t)}) \tag{12}$$

Update time step:

$$t = t + 1 \tag{13}$$

Update the first moment estimate

$$s_t = C_1 s_{t-1} + (1 - C_1) g_t \tag{14}$$

Update the second moment estimate:

$$r_t = C_2 r_{t-1} + (1 - C_2) g_t \odot g_t \tag{15}$$

Correct the deviation of the first-order moment estimation:

$$\hat{s}_t = \frac{s_t}{1 - C_1^t} \tag{16}$$

Correct the deviation of the second moment estimation:

$$\hat{r}_t = \frac{r_t}{1 - C_2^t} \tag{17}$$

Update parameters:

$$\theta_t = \theta_{t-1} - \eta \frac{\hat{s}_t}{\sqrt{\hat{r}_t + \delta}} \tag{18}$$

Compared with other algorithms, the Adam optimization algorithm converges faster, but its convergence effect is not good. It can be seen from the equation (18) in the steps of the Adam algorithm that the learning rate during the training process is affected by the second moment. Therefore, the learning rate oscillation problem may occur during the training process.

5.2. Adam Optimization Algorithm with Memory Factor

We already know that the Adams algorithm adaptively adjusts the learning rate through the first-order moment estimation and the second-order moment estimation. In order to prevent the higher learning rate in the later stage of training, this paper proposes to calculate the long-term average index of adaptive learning rate during the training process.

On the basis of the Adam algorithm, at the beginning of training, we initialize a decay rate ξ_3 of the smooth value exponent of the learning rate, and set the smooth value of the learning rate at the t-th iteration to q_t.

$$q_t = C_3 q_{t-1} + (1 - C_3) \eta_t \tag{19}$$

It can be got that the connection between it and the smooth value of each iteration before the t-th time can be explained as:

$$q_t = (1 - C_3)(q_{t-1} + C_3 q_{t-2} + C_3^2 q_{t-3} + \cdots + C_3^{t-1} q_0) \tag{20}$$

Among them, the decay rate ξ_3 represents a measure of the length of "memory". The closer its value is to 1, the longer the length of "memory".

After calculating the current smoothing value, we choose the smaller value between it and the learning rate calculated by the Adam algorithm as the learning rate for this iteration:

$$\hat{\eta}_t = \min(\eta_t, q_t) \tag{21}$$

In this way, the learning rate of each step can be limited, and as the number of training iterations increases, it can always change in a decreasing trend.

5.3. Adam Optimization Algorithm with Momentum

In the training of the neural network, it selects m samples from the training set at each iteration. These m samples are called a batch. When all the samples are traversed once, it is called an epoch, and at the t-th iteration, the gradient of the loss function L^t with respect to the parameters θ is g_{t0}.

$$g_{t0} = \frac{1}{m} \nabla_\theta \sum L(\theta, x^{(t)}, y^{(t)}) \tag{22}$$

The idea of introducing Nesterov momentum [23] is to add an initial velocity v and momentum parameter α at the beginning. In order to get closer to the correct gradient direction more quickly, Nesterov momentum will add a correction direction after each step of calculating the gradient. The process is as Figure 12.

Therefore, update the parameter θ first,

$$\hat{\theta} = \theta + \alpha v \tag{23}$$

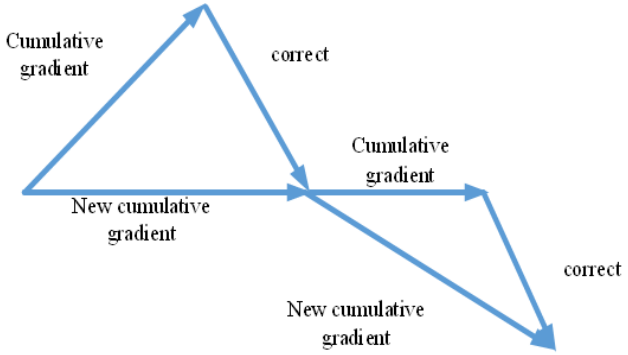


Figure 12. *The correction process of Nesterov momentum.*

According to the updated parameter $\hat{\theta}$, calculate the average gradient g_t of the batch at the t-th iteration,

$$g_t = \frac{1}{m} \nabla_{\hat{\theta}} \sum L(\hat{\theta}, x^{(t)}, y^{(t)}) \tag{24}$$

Then according to equations (15)-(18) in Adam algorithm, we calculate the deviation of the modified first-order moment estimation deviation $\hat{s}_t$. When calculating the modified second-order moment estimation deviation $\hat{r}_t$, this paper chooses the largest one as the current step's modified second moment deviation,

$$\hat{r}_t = \begin{cases} r_t & |r_t| > |\hat{r}_{t-1}| \\ \hat{r}_{t-1} & |r_t| \le |\hat{r}_{t-1}| \end{cases} \tag{25}$$

Then update the velocity,

$$v_t = \alpha v_{t-1} - \eta \frac{\hat{s}_t}{\sqrt{\hat{r}_t + \delta}} \tag{26}$$

Finally update the parameters,

$$\theta_t = \theta_{t-1} + v_t \tag{27}$$

5.4. Improved Adam Optimization Algorithm

The introduction of momentum and memory factor is to control the oscillating learning rate in the later stage of

training. This paper introduces momentum and memory factor into the Adam optimization algorithm at the same time to ensure that the learning rate keeps a monotonically decreasing trend throughout the training process. The steps of the improved algorithm are as follows:

Initialization: learning rateη, initial velocityv, momentum parameter α, initial parameter θ, small constant δ, first-order moment estimation exponential decay rate ξ_1, second-order moment estimation exponential decay rate ξ_2, learning rate smoothing value exponential decay rate ξ_3.

1) Initialization: $s_0 = 0, r_0 = 0, \hat{r}_0 = 0, q_0 = 0, t = 0$;

2) The process of loop iteration:

Select m samples randomly from the training samples: $\{(x^{(1)}, y^{(1)}), \cdots, (x^{(m)}, y^{(m)})\}$;

Update parameters: $\hat{\theta} = \theta + \alpha v$;

Calculate the gradient of this iteration:

$$g = \frac{1}{m} \nabla_{\hat{\theta}} \sum L(\hat{\theta}, x^{(t)}, y^{(t)}) \tag{28}$$

Update number of iterations: $t = t + 1$;

Update the first moment estimate:

$$s_t = C_1 s_{t-1} + (1 - C_1) g_t \tag{29}$$

Update the second moment estimate:

$$r_t = C_2 r_{t-1} + (1 - C_2) g_t \odot g_t \tag{30}$$

Correct the deviation of the first-order moment estimate:

$$\hat{s}_t = \frac{s_1}{1 - C_1^t} \tag{31}$$

Correct the deviation of the second moment estimate:

$$\hat{r}_t = \begin{cases} r_t & |r_t| > |\hat{r}_{t-1}| \\ \hat{r}_{t-1} & |r_t| \le |\hat{r}_{t-1}| \end{cases} \tag{32}$$

Update the smooth value of the learning rate:

$$q_t = C_3 q_{t-1} + (1 - C_3) \eta_t \tag{33}$$

Update learning rate:

$$\hat{\eta}_t = \min(\eta_t, q_t) \tag{34}$$

Update velocity:

$$v = \alpha v - \hat{\eta} \frac{\hat{s}_t}{\sqrt{\hat{r}_t + \delta}} \tag{35}$$

Update parameters:

$$\theta = \theta + v \tag{36}$$

End loop

6. Improved Residual Network

The typical series in the Resnet network are: Resnet18, Resnet34, Resnet50, Resnet101, Resnet150 [24]. Taking into account the hardware conditions of the experimental platform, this paper selects Resnet18 as the experimental network model and makes some changes on the network. The network structure is as follows Figure 13.

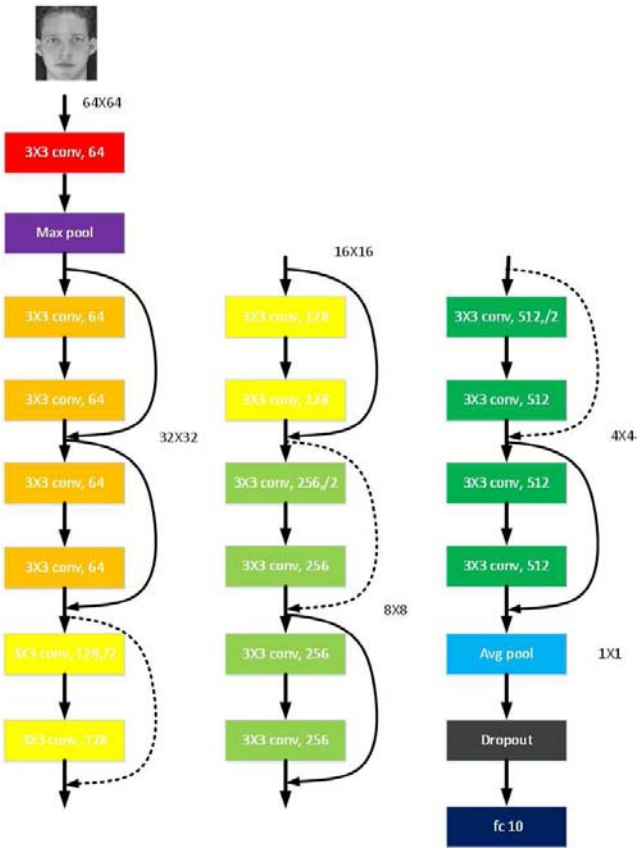


Figure 13. *Structure of Resnet18_new.*

The size of the input face image is 64 × 64. Based on the original Resnet18, this paper changes the size of the convolution kernel of the first convolution layer from 7 × 7 to 3 × 3, and the step size from 2 to 1. Smaller convolution kernel and step size can extract more features of face images. During training, the parameters are initialized as: learning rate η is set to 0.001,ξ_1, ξ_2 and ξ_3 are set to 0.9, 0.999, 0.9, and δ is set to 10-8, respectively, β_1 is set to 8, β_2 is set to 6, and the network selects the relu activation function.

7. Simulation Experiment

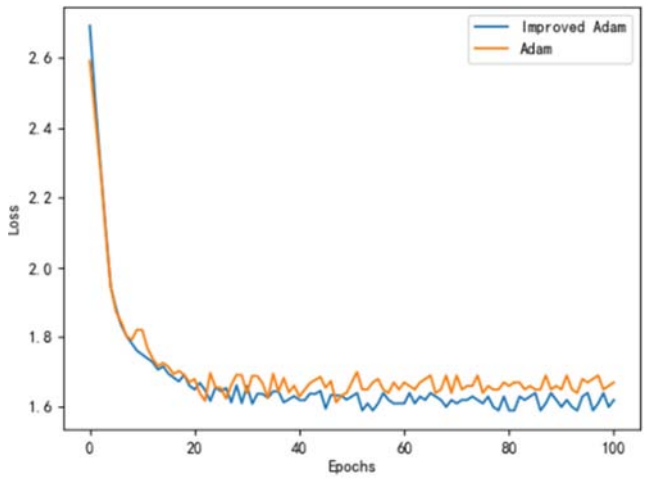
This experimental platform is Windows 10 system, 64-bit, 12G memory, processor is i5-4200H, and graphics card version is NVDIA Geforce 820M. The software platform is python3.6+pycharm3.5, and Tensorflow deep learning framework. In order to prove the effectiveness of this method, this paper selects ORL face database and Yale face database. The ORL face database comes from the AT&T laboratory of the University of Cambridge in the UK, which contains 10 persons, 40 face images of each person, and some samples in the data set are shown in Figure 14. The Yale face database comes from the Visual Control Center of Yale University. It contains 15 persons, each with a total of 11 face images under different lighting, expressions, and poses. Some samples in the data set are shown in Figure 15.

Figure 14. *Some samples in the ORL face database.*

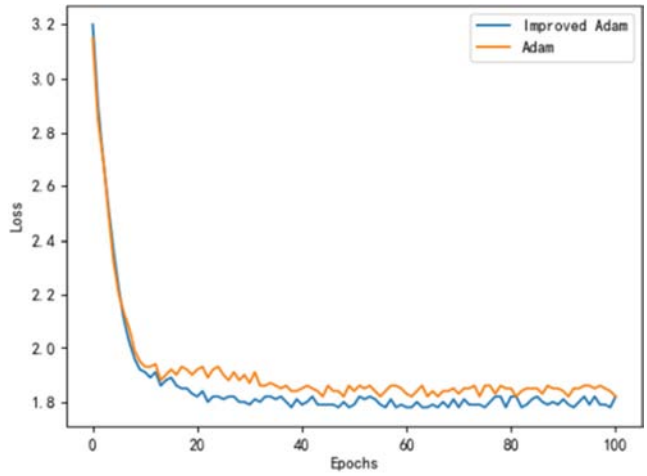
Figure 15. *Some samples in Yale's face database.*

First, this paper conducts a simulation experiment on the ORL face data set after data enhancement, which selects 96 images of each person as the training set, and the number of samples is 960. The remaining 240 face images are used as the test set. When the batch-size is 10, 20, and 40, we compare the performance in the network of the Adam optimization algorithm with the improved Adam optimization algorithm with memory factor and momentum proposed in this paper, We select the joint loss function proposed in this paper as the loss function of the network, and prove the feasibility of this method by comparing the convergence of the loss value of the network on the training set. In order to facilitate the observation of the trend of the curve, we take the average loss of the total number of iterations in each epoch as the loss value of the epoch, and the result is shown in Figure 16. We take the average recognition rate of the total iteration times in each epoch as the recognition accuracy rate of the epoch, and the result is shown in Figure 17.

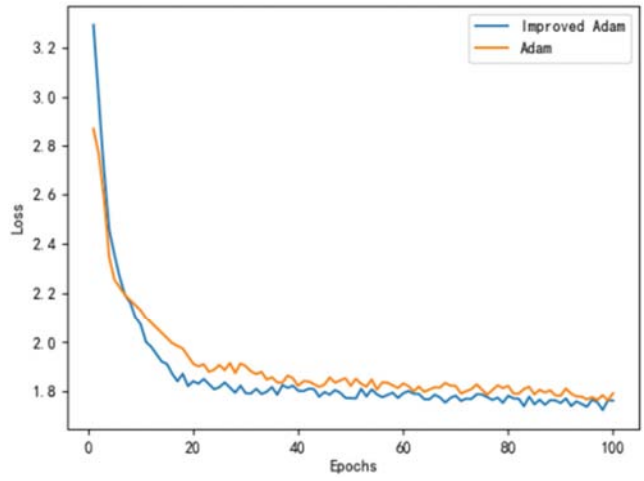
Then, this paper performs the same simulation experiment on the Yale face dataset after data enhancement. Since there are 15 categories of faces in the dataset, we need to change the parameters of the fully connected layer of the network to 15, and select 24 images of each person. The number of training samples is 360, and the remaining 135 images are used as the test set. Similarly, we take the average loss value of all iterations in each epoch as the loss value of the epoch, and the result is shown in Figure 18. The corresponding accuracy rate is shown in Figure 19.



Batch-size=10

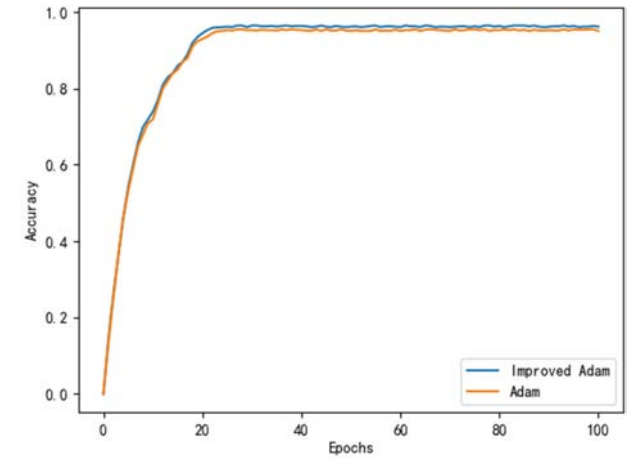


Batch-size=20

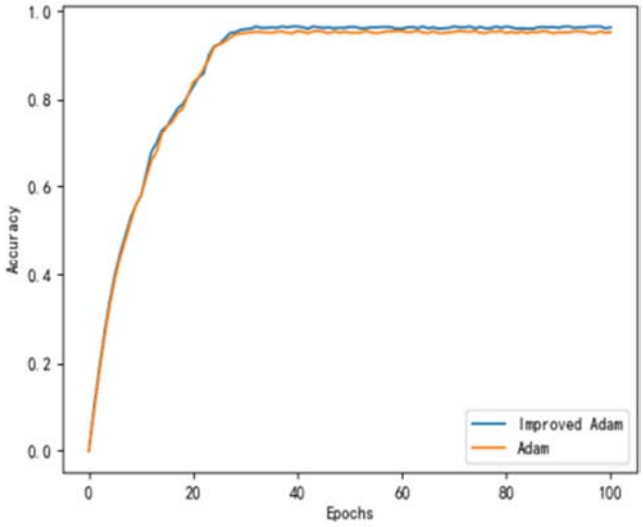


Batch-size=40

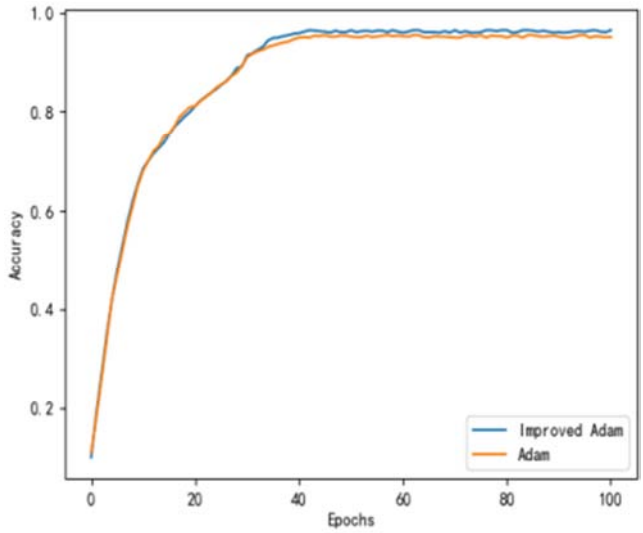
Figure 16. *Loss curve on ORL data set.*



Batch-size=10

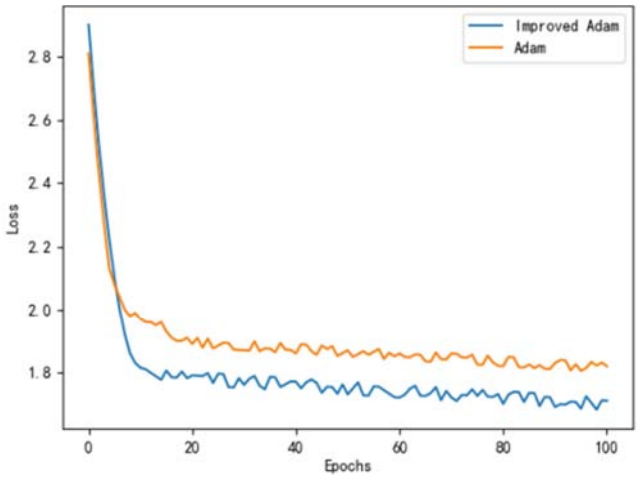


Batch-size=20

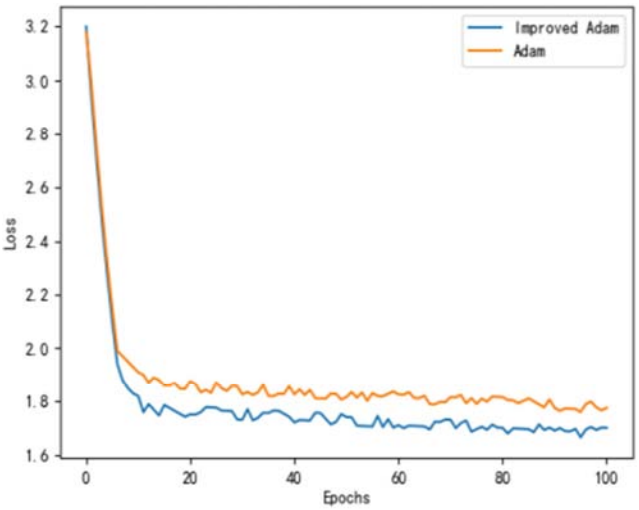


Batch-size=40

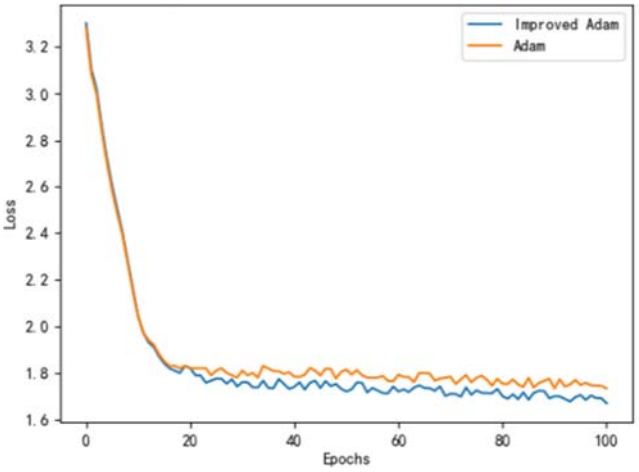
Figure 17. *Accuracy on ORL data set.*



Batch-size=10

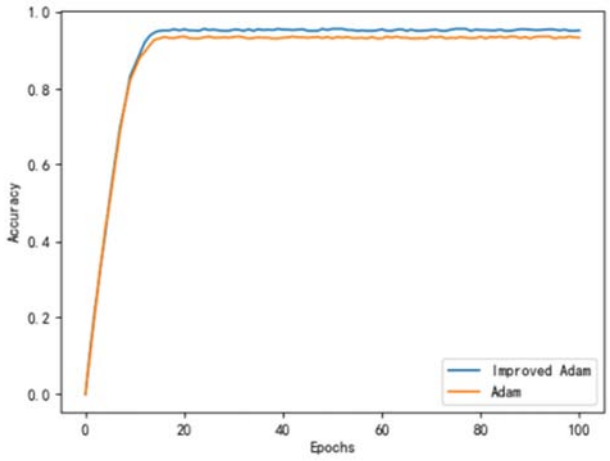


Batch-size=20

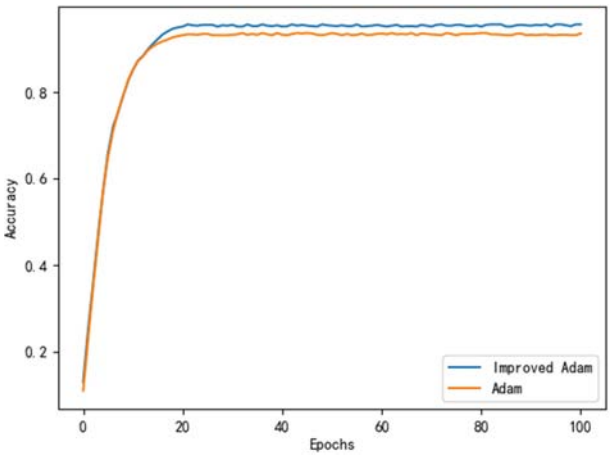


Batch-size=40

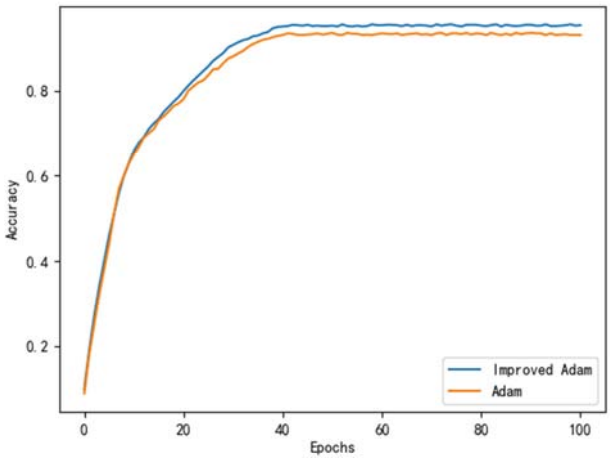
Figure 18. *Loss curve on the Yale data set.*



Batch-size=10



Batch-size=20

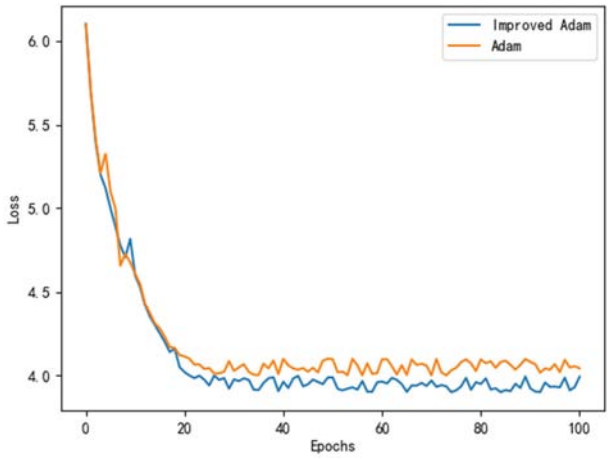


Batch-size=40

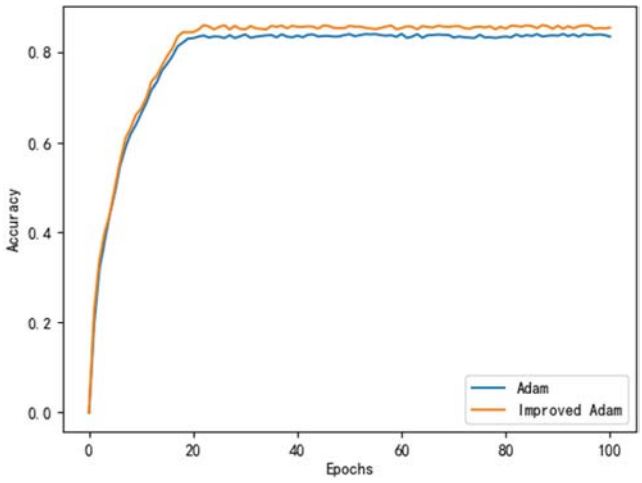
Figure 19. *Accuracy on the Yale data set.*

It can be seen from the results of Figures 16-19 that under different Batch-sizes, the loss value of the residual network of the improved loss function and optimization algorithm on the training set of the ORL face database converges faster than before the improvement And it can converge to a smaller value, the larger the Batch-size, the closer their loss value. The accuracy of face recognition on the test set can be stabilized at the same time before and after the improvement, but the accuracy of the improved algorithm in this paper is increased by about 1%. Similarly, on the Yale face database, the loss value of the improved algorithm in this paper on the training set can also converge to a smaller value, and the accuracy of face recognition on the corresponding test set can be increased by about 2%. There is no "overfitting" in the network, and the convergence of the loss function and the final accuracy of face recognition are more satisfactory.

In order to verify the wide applicability of the improved method proposed in this paper, we conducted simulation experiments on the CMU_PIE face database, because the database contains face images under various illuminations and poses, and the number of samples is more than 40,000. Considering the hardware conditions of the experiment, we selected 2000 face samples under different lighting and postures for the experiment. During training, we set the Batch-size to 20. All samples were traversed 100 times in total, and the number of test samples was 200. The comparison of the results of the loss function convergence and the comparison of the recognition accuracy are shown in Figure 20.



(a) The result of the convergence of the loss function before and after the improvement

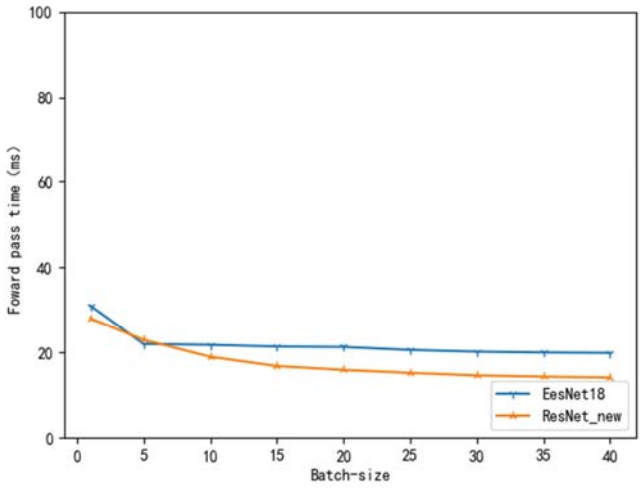


(b) Comparison of recognition accuracy before and after improvement

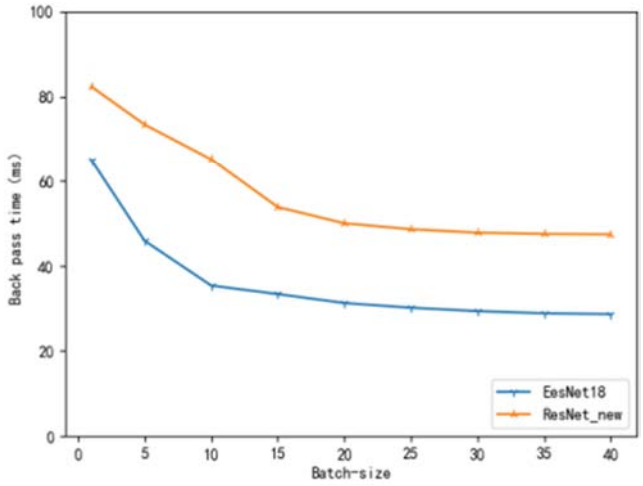
Figure 20. *Comparison of the result of the convergence of the loss function and recognition accuracy of the network before and after the improvement.*

In Figure 20, we can see that on the CMU_PIE database, compared with the original residual network, the improved residual network proposed in this paper has a better loss function convergence effect during the training process. The accuracy of face recognition on the test set is improved by nearly 1%, so it proves that the improved method proposed in this paper has good versatility.

During the experiment of CMU_PIE database, in terms of the running time of the network, this paper compares the forward pass time and the back pass time of the network when training face samples under different batch-sizes. The results are shown in Figure 21.



(a) Forward pass time



(b) Back pass time

Figure 21. *Forward pass time and back pass time of the network before and after the improvement.*

From Figure 21, we can see that with the increase of Batch-size, the time-consuming of forward pass and backward pass are gradually reduced. In the process of forward pass, because we have introduced "dropout" in the improved network, it deactivates some nodes of the network, which makes the network simpler. Therefore, the forward pass consumes less time. However, in the process of back pass, due to the combination of three loss functions and the introduction of memory factor and momentum ideas in the Adam optimization algorithm, this can achieve better convergence and higher face recognition accuracy, But it takes more time. In terms of the total time consumed, the improved network needs more time during the training process.

In terms of algorithm complexity, it is not easy to calculate the complexity of the two networks. Therefore, we analyze their power consumption. We count the average power consumption of the network during the training process under different Batch-sizes. As shown in Figure 22.

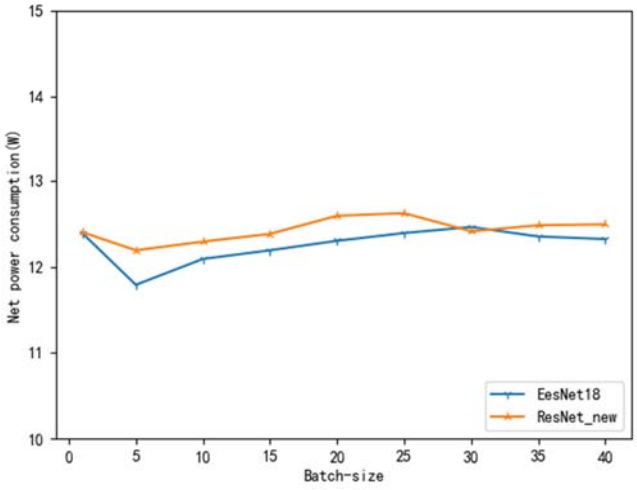


Figure 22. *Comparison of the average power consumption of the network during the training process before and after the improvement.*

From Figure 22, we can see that under the same training conditions, the average power consumption of the improved network is higher, which shows that the complexity of the network has become higher.

8. Conclusion

The method of face sample data enhancement and adding dropout to the network proposed in this paper can solve the problem of "overfitting".. The improved loss function can effectively increase the distance between classes and reduce the distance between classes when classifying faces or other samples. It can achieve the purpose of low coupling and high cohesion. The improved Adam optimization algorithm can prevent the network's learning rate from oscillating to a large extent during the training process, so that the loss value of the network always keeps decreasing and finally converges. Through the results of the final simulation experiment, we can see that when the improved residual network is used in face recognition, the loss value during the sample training process is smaller, and the loss function can reach convergence after fewer iterations. Finally, on the test set, the improved method in this paper improves the recognition accuracy by 1%-2%. Therefore, the improved method proposed in this paper is effective. The disadvantage is that the improved network's running time becomes longer and the algorithm complexity becomes higher.

9. Future Work

This paper solves two problems about the residual network; the first is the problem of "overfitting", and the other is that the loss function of the network does not converge well in the later stage of training. In the future, we still need to focus on how to initialize the network parameters more scientifically. In addition, it is worth studying to use better methods to enhance the face database to solve the problem of network degradation.

Acknowledgements

This work was supported in part by National Natural Science Foundation of China (grant no. 61902268), Sichuan Science and Technology Program (grant no. 2018JY0197, 19ZDZX0037, 2019YFSY0045, 20ZDYF0919, 21ZDYF4052, 2020YFH0124, 2021YFSY0060), Foundation of Deyang Open School-City Cooperative Technology Research and Development (Grant No. 2018CKJSD017), Zigong Key Science and Technology Project of China (2020YGJC01).

References

[1] B. A. White and M. I. Elmasry, "The digi-neocognitron: a digital neocognitron neural network model for VLSI," in *IEEE Transactions on Neural Networks*, vol. 3, no. 1, pp. 73-85, Jan. 1992.

[2] D. J. Toms, "Training binary node feedforward neural networks by back propagation of error," in *Electronics Letters*, vol. 26, no. 21, pp. 1745-1746, 11 Oct. 1990.

[3] J. Lin, C. Chao and J. Chiou, "Determining Neuronal Number in Each Hidden Layer Using Earthquake Catalogues as Training Data in Training an Embedded Back Propagation Neural Network for Predicting Earthquake Magnitude," in *IEEE Access*, vol. 6, pp. 52582-52597, 2018.

[4] W. Kehe, Z. Ying and G. Minghao, "Situation Awareness Technology of LeNet-5 Attack Detection Model Based on Optimized Feature Set," *2020 IEEE 11th International Conference on Software Engineering and Service Science (ICSESS)*, Beijing, China, 2020, pp. 269-272.

[5] Y. Guo, Z. Pang, J. Du, F. Jiang and Q. Hu, "An Improved AlexNet for Power Edge Transmission Line Anomaly Detection," in *IEEE Access*, vol. 8, pp. 97830-97838, 2020.

[6] S. Sun *et al*., "Fault Diagnosis of Conventional Circuit Breaker Contact System Based on Time–Frequency Analysis and Improved AlexNet," in *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1-12, 2021, Art no. 3508512.

[7] X. Yao, X. Wang, Y. Karaca, J. Xie and S. Wang, "Glomerulus Classification via an Improved GoogLeNet," in *IEEE Access*, vol. 8, pp. 176916-176923, 2020.

[8] L. Balagourouchetty, J. K. Pragatheeswaran, B. Pottakkat and R. G, "GoogLeNet-Based Ensemble FCNet Classifier for Focal Liver Lesion Diagnosis," in *IEEE Journal of Biomedical and Health Informatics*, vol. 24, no. 6, pp. 1686-1694, June 2020.

[9] H. Ke, D. Chen, X. Li, Y. Tang, T. Shah and R. Ranjan, "Towards Brain Big Data Classification: Epileptic EEG Identification With a Lightweight VGGNet on Global MIC," in *IEEE Access*, vol. 6, pp. 14722-14733, 2018.

[10] A. Mahajan and S. Chaudhary, "Categorical Image Classification Based On Representational Deep Network (RESNET)," 2019 3rd *International conference on Electronics, Communication and Aerospace Technology (ICECA)*, Coimbatore, India, 2019, pp. 327-330.

[11] H. Zhu, N. Lin, H. Leung, R. Leung and S. Theodoidis, "Target Classification From SAR Imagery Based on the Pixel Grayscale Decline by Graph Convolutional Neural Network," in *IEEE Sensors Letters*, vol. 4, no. 6, pp. 1-4, June 2020, Art no. 7002204.

[12] C. Ding and D. Tao, "Trunk-Branch Ensemble Convolutional Neural Networks for Video-Based Face Recognition," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 4, pp. 1002-1014, 1 April 2018.

[13] S. Srisuk and S. Ongkittikul, "Robust face recognition based on weighted DeepFace," 2017 *International Electrical Engineering Congress (iEECON)*, Pattaya, 2017, pp. 1-4.

[14] S. Y. Wong, K. S. Yap, Q. Zhai and X. Li, "Realization of a Hybrid Locally Connected Extreme Learning Machine With DeepID for Face Verification," in *IEEE Access*, vol. 7, pp. 70447-70460, 2019.

[15] E. Jose, G. M., M. T. P. Haridas and M. H. Supriya, "Face Recognition based Surveillance System Using FaceNet and MTCNN on Jetson TX2," 2019 5th *International Conference on Advanced Computing & Communication Systems (ICACCS)*, Coimbatore, India, 2019, pp. 608-613.

[16] Y. Liu, J. A. Starzyk and Z. Zhu, "Optimized Approximation Algorithm in Neural Networks Without Overfitting," in *IEEE Transactions on Neural Networks*, vol. 19, no. 6, pp. 983-995, June 2008.

[17] M. A. Abuzneid and A. Mahmood, "Enhanced Human Face Recognition Using LBPH Descriptor, Multi-KNN, and Back-Propagation Neural Network," in *IEEE Access*, vol. 6, pp. 20641-20651, 2018.

[18] Y. Zheng, Z. Gao, Y. Wang and Q. Fu, "MOOC Dropout Prediction Using FWTS-CNN Model Based on Fused Feature Weighting and Time Series," in *IEEE Access*, vol. 8, pp. 225324-225335, 2020.

[19] W. Zhang, Y. Chen, W. Yang, G. Wang, J. Xue and Q. Liao, "Class-Variant Margin Normalized Softmax Loss for Deep Face Recognition," in *IEEE Transactions on Neural Networks and Learning Systems*.

[20] C. Qi and F. Su, "Contrastive-center loss for deep neural networks," 2017 *IEEE International Conference on Image Processing (ICIP)*, Beijing, 2017, pp. 2851-2855.

[21] W. Lei, R. Zhang, Y. Yang, R. Wang and W. Zheng, "Class-Center Involved Triplet Loss for Skin Disease Classification on Imbalanced Data," 2020 *IEEE 17th International Symposium on Biomedical Imaging (ISBI)*, Iowa City, IA, USA, 2020, pp. 1-5.

[22] Z. Zhang, "Improved Adam Optimizer for Deep Neural Networks," 2018 IEEE/ACM 26th *International Symposium on Quality of Service (IWQoS)*, Banff, AB, Canada, 2018, pp. 1-2.

[23] X. Li, W. Wang, S. Zhu, W. Xiang and X. Wu, "Generalized Nesterov Accelerated Conjugate Gradient Algorithm for a Compressively Sampled MR Imaging Reconstruction," in *IEEE Access*, vol. 8, pp. 157130-157139, 2020.

[24] B. Li and Y. He, "An Improved ResNet Based on the Adjustable Shortcut Connections," in *IEEE Access*, vol. 6, pp. 18967-18974, 2018.

Biography

Tang Xiaolin was born in Dazhou City, Sichuan Province, China in 1994. He received a bachelor's degree from Sichuan University of Science and Engineering, and now he is a graduate student in the school of Automation and Information Engineering of Sichuan University of Science and Engineering. His main research direction is intelligent control and system optimization.

Hou Jin was born in Zigong, Sichuan, China in 1968. He received the bachelor degree in process automation from Sichuan University of Science and Engineering (SUSE) in 1989, and the master degree in chemical engineering from Sichuan University in 2003. Before 2017, he was the chief engineer in automation with a large chemical corporateand was appointed as professorate senior engineer in 2011.

Huang Ye was born in Xichang City, Sichuan Province, China in 1993. He received a bachelor's degree from Sichuan University of Science and Engineering, and now he is a graduate student in the school of Automation and Information Engineering of Sichuan University of Science and Engineering. His main research direction is intelligent control and system optimization.

Wang Xiaogang was born in Baoji City, Shanxi Province, China in 1984. He received his Ph. D degree from the Chongqing University, Chongqing, China. He is currently an associate professor, with Artificial Intelligence Key Laboratory of Sichuan Province, School of Automation and Information Engineering, Sichuan University of Science and Engineering. His current interests are in the area of wireless sensor network and security, IoT, artificial intelligence.

Han Yiting was born in Zigong City, Sichuan Province, China in 1996. She received a bachelor's degree from Chengdu Technological University and now she is a graduate student in the school of Automation and Information Engineering of Sichuan University of Science and Engineering. Her main research direction is intelligent control and system optimization.