Mathematical Biosciences and Engineering

MBE, 16(6): 8309–8321.
DOI: 10.3934/mbe.2019420
Received: 24 April 2019
Accepted: 25 August 2019
Published: 17 September 2019

http://www.aimspress.com/journal/MBE

Research article

An effective classifier based on convolutional neural network and regularized extreme learning machine

Chunmei He*, Hongyu Kang, Tong Yao and Xiaorui Li

College of Information Engineering, Xiangtan University, Xiangtan, Hunan 411105, China

*** Correspondence:** Email: xiaoxiao_he8@163.com.

Abstract: An effective classifier combining convolutional neural network and regularized extreme learning machine (called as CNN-RELM) is presented in this paper. Firstly, CNN-RELM trains the convolutional neural network (CNN) using the gradient descent method until the learning target accuracy reaches. Then the fully connected layer of CNN is replaced by regularized extreme learning machine (RELM) optimized by genetic algorithm and the rest layers of the CNN remain unchanged. The experiments on different face databases are given to evaluate the performance of CNN-RELM. The experimental results show that CNN-RELM is a feasible classifier and it outperforms CNN and RELM. Due to the uniting of CNN and RELM, CNN-RELM have the advantages of CNN and RELM and it is easier to learn and faster in testing.

Keywords: convolutional neural network; regularized extreme learning machine; classification; face recognition; feature extraction

1. Introduction

Deep learning becomes one of the most important technologies in the field of artificial intelligence [1–3]. CNN is the most representative model of deep learning [4]. In 1989, Yann LeCun [5] proposes CNN model and applies CNN to handwritten character recognition. Yoshua Bengio presents the Probabilistic models of sequences and proposes generative adversarial networks [6]. Geoffrey Hinton et al. present a deep brief net [7] and apply CNN in Image Net game. The advantages of CNN against traditional methods can be summarized as follows [4].

(1) Hierarchical feature representation.

(2) Compared with traditional shallow models, a deeper architecture provides an exponentially increased expressive capability.
(3) The architecture of CNN provides an opportunity to optimize several related tasks together.
(4) Benefiting from the large learning capacity of CNNs, some classical computer vision challenges can be recast as high-dimensional data transform problems and solved from a different viewpoint.

Due to these advantages, CNN has been widely applied into many research fields [8–14]. However, the CNN has some disadvantages. Many Parameters need to be adjusted in CNN. It needs many training samples and the GPU is preferred for training CNN.

Extreme learning machine (ELM) [15] is a kind of feed-forward neural network in essence. ELM generates input layer weights and bias values at random, directly solves the least square solution of the output weights, and obtains the final training model at the same time. ELM has the advantages such as no iterative calculation, fast learning speed and strong generalization ability. Many scholars pay close attention to ELM and have achieved good results [16–20]. A fast kernel ELM combining the conjugate gradient method (CG-KELM) is presented in [16] and the kernel ELM is applied in image restoration. In [17], the paper further studies ELM for classification in the aspect of the standard optimization method and extends ELM to a specific type of "generalized" single-hidden layer feedforward networks-support vector network. An improved meta-learning model of ELM is proposed in [18]. Chunmei He et al. [19] propose a fast learning algorithm for the regular fuzzy neural network based on ELM which has good performance and approximation ability. Since ELM is based on the principle of empirical risk minimization which may lead to over-fitting problem. In [20], the scholars combine the structural risk minimization theory and weighted least square method and present RELM. RELM considers both the structural risk and the empirical risk and balances the two risks using the risk ratio parameter. RELM is widely used in the fields of classification, regression and prediction. RELM has the advantage of fast learning speed and further improves the generalization performance of ELM.

Motivated by the remarkable success of CNN and RELM, RELM is introduced into CNN and an effective classifier CNN-RELM is proposed in this paper. In CNN-RELM, CNN can extract the deep feature of the input, while RELM have fast learning speed and good generalized ability to acquire better recognized accuracy. The rest paper is organized as follows. Section 2 introduces the basic theory of CNN, ELM and RELM. The CNN-RELM model and learning algorithm are presented in section 3. The experiment simulations show the excellent performance of the CNN-RELM in section 4. Section 5 summarizes the study and discusses the future work.

2. Brief reviews of CNN, ELM and RELM

2.1. Convolutional Neural Network

A typical CNN is simply introduced here. The CNN topological model is shown in Figure 1.

In the CNN, the convolution layer and pooling layer extract the features and input into the fully connected layer to obtain the classification results. The feature extraction process is as follows.

In the convolution layer, convolution is performed on the input, and the output is

$$x_j^l = f\left(\sum_{i \in M_j} x_i^{l-1} * w_{ij}^l + b_j^l\right) \tag{1}$$

where M_j is the input set and b is the bias of the convolution layer.

The gradient of the convolution layer is defined by $\delta_j^l = \beta_j^{l+1}\left(f'\left(u_j^l\right) \circ up\left(\delta_j^{l+1}\right)\right)$.

The output of the pooling layer is

$$\mathrm{x}_j^l = f\left(\beta_j^l down\left(x_i^{l-1}\right) + b_j^l\right) \tag{2}$$

where $down(\cdot)$ is the pooling function, b is the bias, and β is the weight.

In the fully connected layer, the output of the l-th layer is $\mathrm{x}^l = f\left(u^l\right)$, where $\mathrm{u}^l = w^l x^{l-1} + b^l$, w^l is the weight and b^l is the bias of the l-th layer.

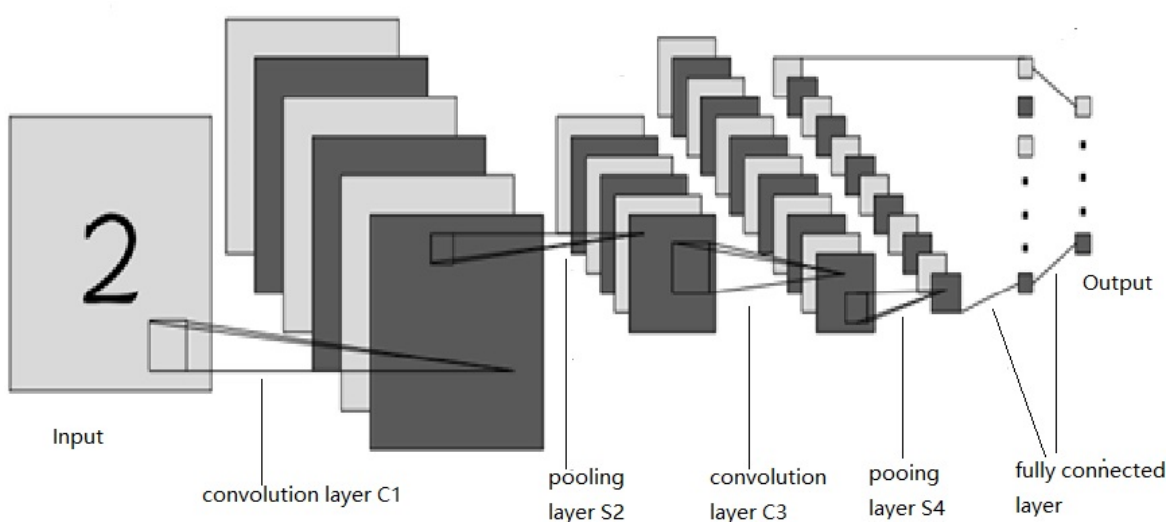

Figure 1. CNN topological model. The CNN has two convolution layers C_1 and C_3, two pooling layers S_2 and S_4 and the fully connected layer.

2.2. ELM and RELM

We simply introduce ELM and RELM. For more details, please refer to [15,20]. The topology of ELM is shown in Figure 2.

Different from other feed-forward neural networks, the input weight w_i and bias b_i of the ELM are generated randomly in the training. After the input sample set (x_j, y_j) is processed by the hidden layer neurons, the hidden layer output matrix H of ELM can be fixed. The goal of ELM is to adjust the weight $\hat{\beta}$ satisfying the following equation.

$$\left\|\mathrm{H} \cdot \hat{\beta} - \mathrm{Y}\right\| = \min_{\beta}\left\|\mathrm{H} \cdot \beta - \mathrm{Y}\right\| \tag{3}$$

where $\beta = \left[\beta_1^T, \cdots, \beta_l^T\right]_{l \times m}^T$, $\mathrm{Y} = \left[y_1^T, \cdots, y_N^T\right]_{N \times m}^T$ is the expected output and

$$\mathrm{H}(w_1,\cdots,w_l,b_1,\cdots,b_l,x_1,\cdots,x_n)=\begin{bmatrix} \mathrm{g}(w_1\cdot x_1+b_1)\cdots\mathrm{g}(w_l\cdot x_1+b_l) \\ \vdots\cdots\vdots \\ \mathrm{g}(w_1\cdot x_n+b_1)\cdots\mathrm{g}(w_l\cdot x_n+b_l) \end{bmatrix}_{l\times n}. \tag{4}$$

The solution of Eq (3) is the least normal square solution as follows.

$$\hat{\beta} = \mathrm{H}^{+}\mathrm{Y}, \tag{5}$$

where H^{+} is the Moore-Penrose generalized inverse matrix.

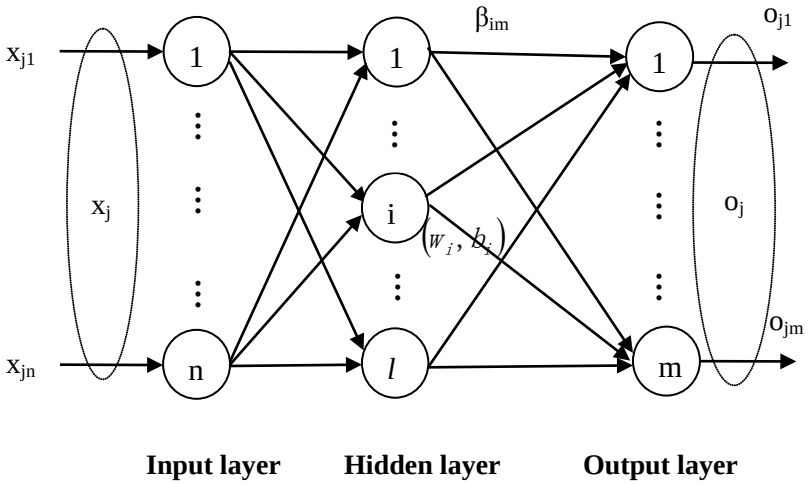

Figure 2. The topology of ELM. In Figure 2, ELM has n input layer nodes, l hidden layer nodes and m output nodes. $x_j=[x_{j1},x_{j2},\cdots,x_{jn}]^T\in R^n$ is the input, w_i is the weight, b_i is the bias, $o_j=[o_{j1},o_{j2},\cdots,o_{jm}]^T\in R^m$ is the actual output and β is the output weight matrix between the hidden layer and the output layer.

ELM only considers the empirical risk and doesn't consider the structural risk. ELM directly calculates the least squares solution, and users can't make fine-tuning according to the characteristics of the database, these results in poor control ability and over-fitting problems. Therefore, the structural risk minimization theory is introduced into ELM, and RELM is proposed [20]. The RELM model is as follows. When $\gamma\to\infty$, the RELM degenerates into the ELM, that is, ELM is a special case of the RELM.

$$\arg_\beta\min \mathrm{E}(W)=\arg_\beta\min \mathrm{E}\left(0.5\|\beta\|^2+0.5\gamma\|\varepsilon\|^2\right),$$

$$s.t.\sum_{i=1}^{l}\beta_i\mathrm{g}(w_i\cdot x_j+b_i)-y_j=\varepsilon_j,\ j=1,\cdots,n. \tag{6}$$

where $\|\beta\|^2$ is the structural risk, ε is the error, $\|\varepsilon\|^2$ is the empirical risk, γ is the proportion parameters to balance empirical risk and structural risk. The Eq (6) is a conditional extreme problem, which is solved by converting Lagrange equation into an unconditional extreme problem as follows.

$$\begin{aligned}\ell(\beta,\varepsilon,\alpha)&=\frac{\gamma}{2}\|\varepsilon\|^2+\frac{1}{2}\|\beta\|^2-\sum_{j=1}^{n}\alpha_j\left(g\left(w_i\cdot x_j+b_j\right)-y_j-\varepsilon_j\right)\\&=\frac{\gamma}{2}\|\varepsilon\|^2+\frac{1}{2}\|\beta\|^2-\alpha(\mathrm{H}\beta-\mathrm{Y}-\varepsilon)\end{aligned}\tag{7}$$

where $\alpha_j\in R^m(j=1,\cdots,n)$ is Lagrange operator. Let the gradient of the Lagrange equation be 0 and compute the weight β as follows.

$$\beta=\left(\frac{I}{\gamma}+\mathrm{H}^{\mathrm{T}}\mathrm{H}\right)^{+}\mathrm{H}^{\mathrm{T}}\mathrm{Y}.\tag{8}$$

3. An effective classifier: CNN-RELM

In this section, the CNN-RELM model and learning algorithm are presented. In CNN-RELM, the convolutional hidden layer and the pooling layer in CNN extract deep features from the original input and then RELM are used for feature classification. The CNN-RELM model is changed in the two steps of training period. And in testing period, there is no fully connected layer in CNN which is replaced by the RELM.A trained CNN-RELM used for testing is as in Figure 3. The training of CNN-RELM model is divided into two steps: The training of CNN and the merger of CNN and RELM. These two steps are presented in detail as follows.

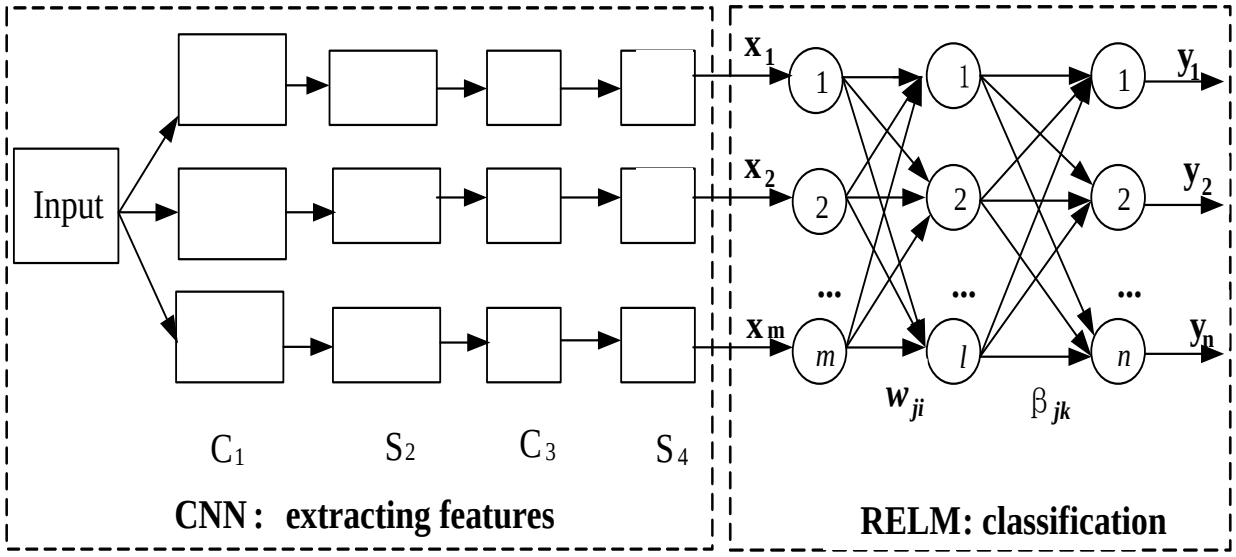

Figure 3. The trained CNN-RELM for testing. In CNN, input is the sample image, C_1 and C_3 are the convolutional layers, S_2 and S_4 are the pooling layers. In RELM, x is the input, w is the weight between input-layer and hidden-layer, β is the weight between hidden-layer and output-layer, and y is the output.

3.1. The training of CNN

In this step, the CNN model is trained. The CNN model in this step is as in Figure 4. The related parameters in CNN are adjusted by the gradient descent method according to the errors between the actual output and the expected output. The training of the CNN stops if the minimum error reaches or the maximum number of iterations reaches. Then the CNN is saved for the next step.

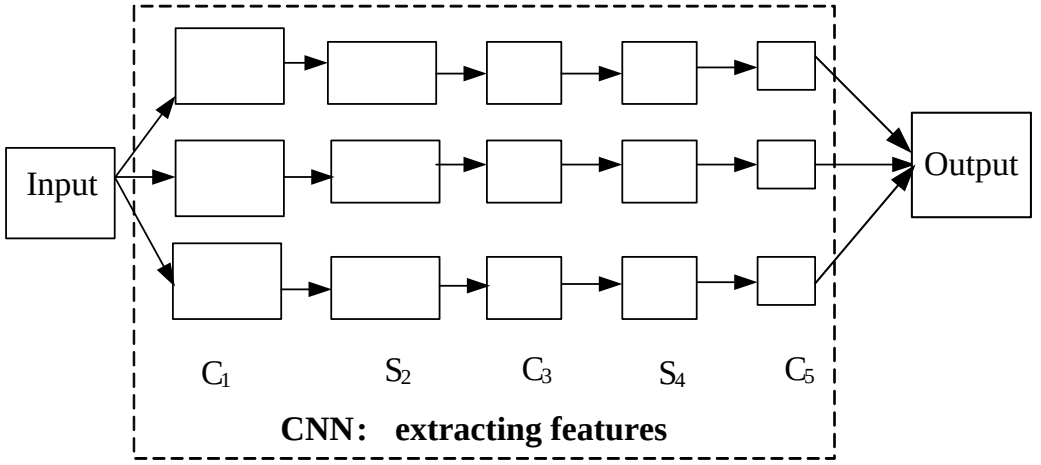

Figure 4. The topology of CNN model. In CNN, C_1 and C_3 are two convolution layers, S_2 and S_4 are two pooling layers, and C_5 is the fully connected layer.

The feature map in the CNN is adjusted as follows.

(a) If the m-th layer is a convolution layer, the n-th feature map is

$$x_j^l = f\left(\sum_{x_i^{m-1} \in M_n} x_i^{m-1} * k_{in}^m + b_n^m\right) \tag{9}$$

where M_n is the input set; f is a nonlinear active function; k_{in}^m is the convolution kernel; b_n^m is bias.

(b) If the m-th layer is a pool layer, its n-th feature map is

$$x_n^m = f\left(w_n^m down\left(x_n^{m-1}\right) + b_n^m\right) \tag{10}$$

where w_n^m is weight, b_n^m is bias and $down(\cdot)$ is pooling function. The two pooling methods: The max pooling and the mean pooling are used here.

3.2. The merger of CNN and RELM

In this step, we firstly acquire the RELM which is optimized by genetic algorithm. And then the convolutional layers and pooling layers of the CNN is fixed while the full-connected layer of the CNN is replaced by the RELM. The topology of RELM is also shown as in Figure 3. The optimal risk ratio parameters γ in the RELM are optimized by genetic algorithm. The RELM mathematical model is the same as Eq (6). The weight β is computed by Eq (8). The fully-connected layer of

CNN is replaced by the RELM. In other words, the feature images trained by CNN are considered as the input of RELM, and the desired classification results are obtained through RELM. The role of RELM is to act as a classifier in CNN-RELM model.

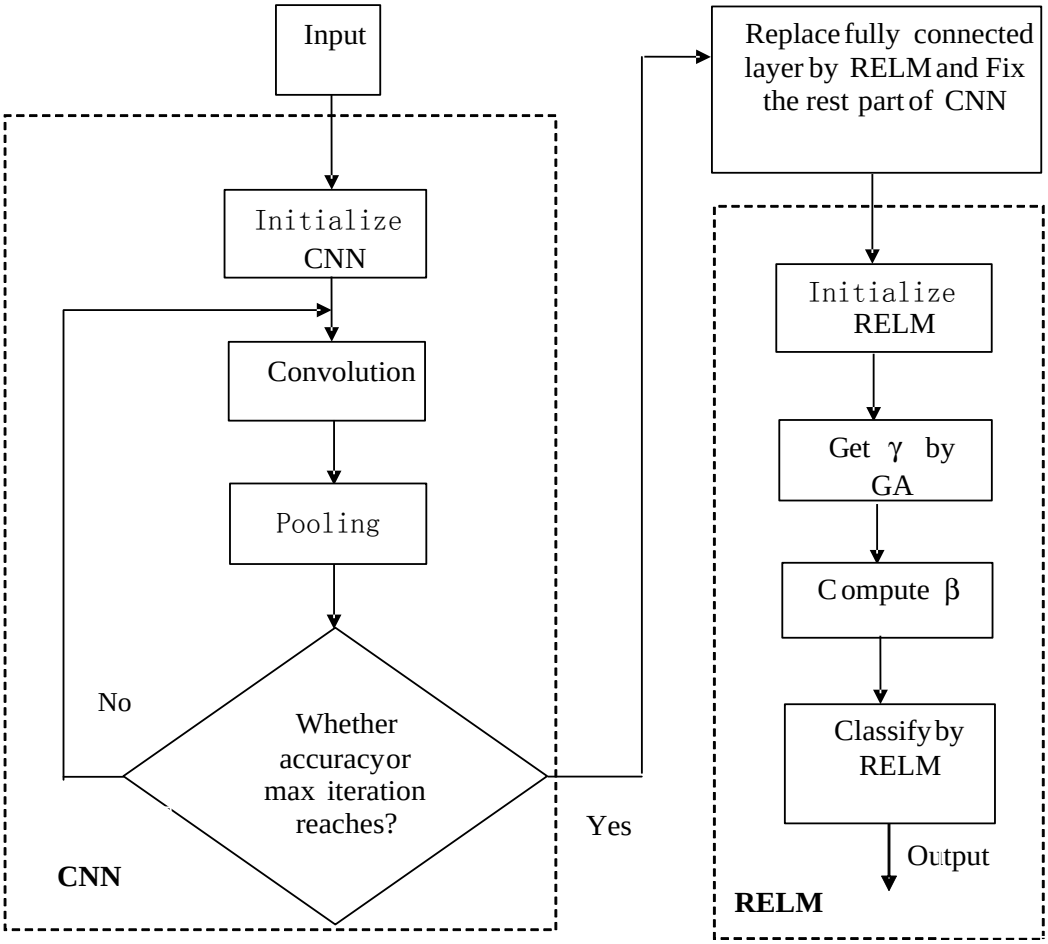

Figure 5. The learning algorithm flow chart of CNN-RELM. CNN-RLEM is divided into two parts: CNN and RELM. The related parameters in CNN are adjusted by the gradient descent method according to the errors between the actual output and the expected output. The training process stops if the minimum error reaches or the maximum number of iterations reaches. Then the main part of the CNN is fixed except that the full-connected layer of the CNN is replaced by RELM. The optimal risk ratio parameters γ in the RELM are optimized by genetic algorithm.

3.3. The learning algorithm of CNN-RELM

The learning algorithm of CNN-RELM is outlined in Algorithm 1. The learning algorithm flow chart of CNN-RELM is shown as in Figure 5.

Algorithm 1

Step 1. Initialize the CNN parameters, expected target and maximum iteration times.

Step 2. Compute the actual output of the network by Eqs (9), (10).

Step 3. If the predetermined target precision reaches or the maximum number of iterations reaches, then go to Step 5, else go to Step 4.

Step 4. Adjust the parameters of CNN by the gradient descent method and go to Step 2.

Step 5. Initialize RELM. Randomly initialize the weights and biases. Get the regularized parameters by GA.

Step 6. Let the feature vectors obtained by CNN into RELM and Compute β by Eq (8).

Step 7. Save the CNN-RELM and classify by the CNN-RELM.

4. Simulation experiments

In this section, we present the face recognition experiment to verify the feasibility of CNN-RELM and compare it with RELM and CNN. The impacts of different pooling methods and different number of training samples on the performance of CNN-RELM are also presented.

4.1. Face databases and experiment environments

Two face databases: ORL and NUST are used in the simulations. The detailed database information is shown in Table 1. The ORL face database is founded by the Olivetti research laboratory in Cambridge, England. The ORL face database can be downloaded in the following website: http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html. The NUST face database is made by Nanjing University of Science and Technology, China.

Table 1. Face databases information.

databases	classes	features	number of samples	remarks
ORL	40	10	400	These include changes in facial expressions, minor changes in posture, and changes in scale within 20%.
NUST	96	10	216	It mainly includes the change of face's different pose.

The experimental execution environment is as follows. Software environment: Matlab-R2014a. Operation system: Windows 8. Hardware environment: Dell PC. CPU: Intel(R) Core(TM) i5-6500 CPU @ 3.20GHz. Hard disk: Seagate ST500DM002-1SB10A (500 GB /7200 rpm). Memory: 32 GB (Samsung DDR4 2666MHz).

4.2. Face recognition experiment by CNN-RELM

To evaluate the performance of CNN-RELM, some simulations on the ORL face databases are given. In training of the CNN, there are two convolution layers: C_1, C_3 and two pool layers: S_2, S_4. The convolution kernel of convolution layer C_1 and C_3 are all set as 9×9 matrix. In pooling

layer S_2 and S_4, max-pooling is used and the window size is 3×3. The gradient descent algorithm is used to train the CNN. The error function of CNN in the training is shown in Figure 6. As the number of iterations increases, the training error gradually decreases.

After the CNN is trained, the fully connected layer is replaced by RELM for classification. The feature images obtained by the pooling layer S_4 are converted into column vectors and stored in matrix. Moreover, the RELM model is used for classification. When initializing RELM, we obtain the input weight and bias in a random way. At the same time, we use the optimal risk ratio parameter mentioned in [21], and the hidden layer nodes are set to 200. The test results are shown in Figure 7. The experimental results show that the proposed algorithm is feasible in face recognition.

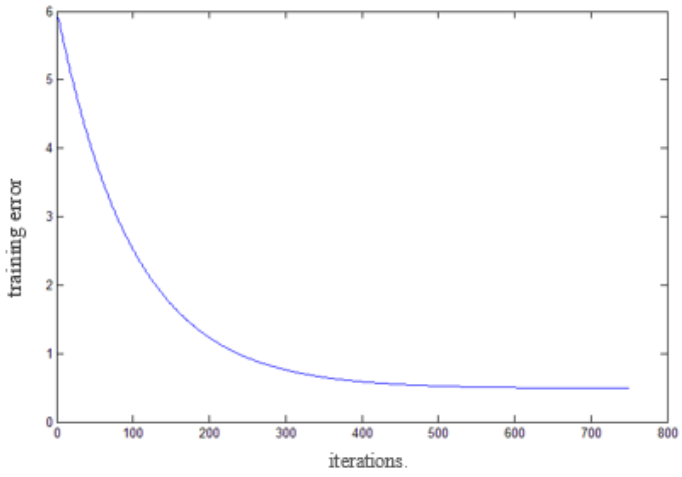

Figure 6. Training error curve of CNN.

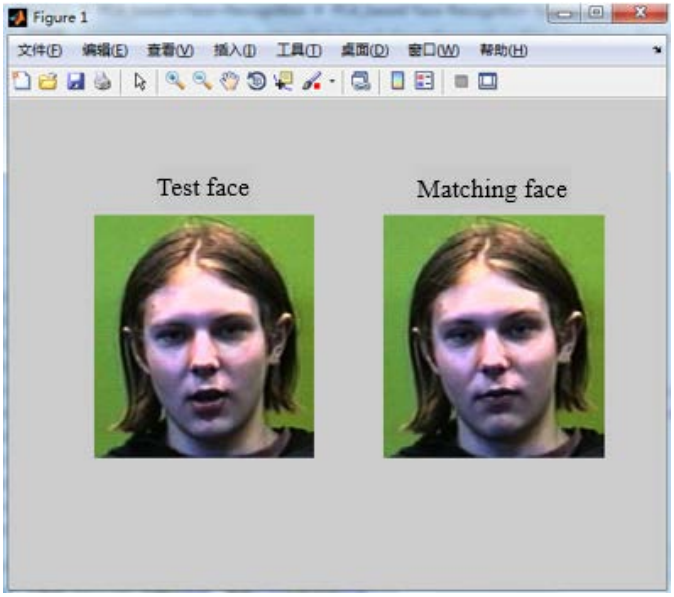

Figure 7. Face recognition results of CNN-RELM.

4.3. Comparison experiments to other methods

In this section, we compare the performances of CNN-RELM, RELM and CNN in face recognition experiments. The parameters in the CNN and the risk ratio parameter of RELM are similar to that in [21]. Two standard face databases: ORL and NUST as in Table 1 are used in the comparative experiments. In the experiment, we select seven images of each person as training samples, and the remaining three images as test samples. All hidden layer node number of neural networks is set to 200. The recognition results of the three algorithms are shown in Table 2.

As seen from Table 2, the CNN-RELM model proposed in this paper has the best recognition accuracy. In next section, an experiment is given to study the influence of different pooling methods and number of the training sample on CNN-RELM.

Table 2. Comparison of different methods in face recognition. Specially point out: The bold fonts are the results of our method.

databases	methods	division of sample sets		classification accuracy (%)
		training samples	test samples	
ORL	RELM	280	120	91.67
	CNN	280	120	90.00
	CNN-RELM	280	120	**96.67**
NUST	RELM	672	288	89.58
	CNN	672	288	88.54
	CNN-RELM	672	288	**96.88**

4.4. Impact of different pooling methods and number of training samples on the performance of CNN-RELM

In order to evaluate how different pooling methods and different training samples impact on the performance of the CNN-RELM, the experiment is conducted in this section. The ORL face database is selected here. The parameters of the CNN-RELM model are selected as the same as those in section 4.2. The pooling method respectively uses Max-pooling, Average-pooling, Stochastic-pooling and Lp-pooling. Figures 4–7 are selected for each class's training samples and the rest pictures are taken as test samples. The experimental results are shown in Table 3 and Figure 8.

4.5. Discussion and result analysis

By the experiment in Figures 6 and 7, it’s known that the CNN-RELM is feasible in classification. From Table 2, we can concluded that the CNN-RELM outperform CNN and RELM in classification. The CNN-RELM model combines CNN with RELM and overcomes the deficiency of the two models. The CNN-RELM can also be used in other application tasks such as remote sensing and object shape reconstruction and the process is similar to that of classification.

From Table 3 and Figure 8, we can see that the recognition rate increase at the same pooling strategy as the selected training samples increase. When the same number of training samples is

selected, the L_p – pooling strategy has the highest recognition rate. The recognition rate corresponding to the average pooling strategy, the statistics pooling strategy and the maximum pooling strategy are sequentially reduced.

Seen from Figure 8, the selection of different pooling methods has an impact on the performance of CNN-RELM. In practical applications, the selection of appropriate pooling methods according to the actual situation of data is conducive to achieving better application results.

Table 3. Face recognition results of CNN-RELM with different pooling methods and different number of training samples. The highest recognition rate marks in bold fonts.

methods	training samples			
	4	5	6	7
Max-pooling	83.75	88.00	94.37	96.67
Average-pooling	84.17	92.00	96.25	97.50
Stochastic-pooling	83.75	89.50	95.00	96.67
Lp-pooling	**84.58**	**92.50**	**96.88**	**98.33**

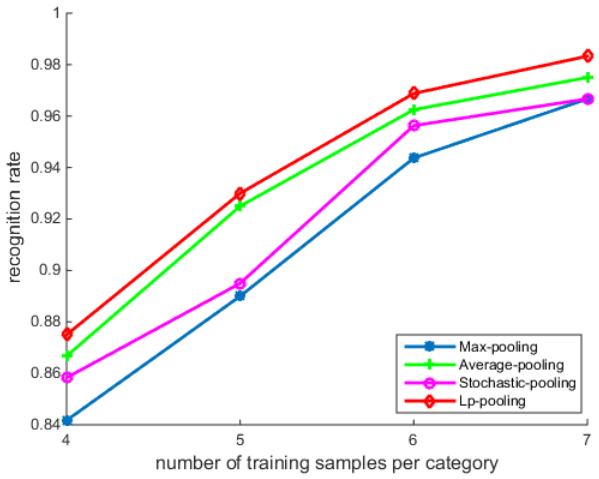

Figure 8. Face recognition results of different pooling methods. In the figure, the abscissa represents the number of training samples for each category, and the ordinate is the corresponding recognition rate. For example, the abscissa value 4 indicates that 4 pictures are selected as training samples and the remaining ones are used as test samples. The ordinate is the corresponding recognition rate.

5. Conclusions

An effective classifier CNN-RELM is proposed in this paper. Firstly, the CNN-RELM trains the convolutional neural network using the gradient descent method until the learning target accuracy reaches. Then the fully connected layer of CNN is replaced by RELM optimized by genetic algorithm and the rest layers of the CNN remain unchanged. A series of experiments conducted on ORL and NUST databases show that the CNN-RELM outperforms CNN and RELM in classification

and demonstrate the efficiency and accuracy of the proposed CNN-RELM model. Meanwhile, we also verify that the selection of different pooling methods has an impact on the performance of CNN-RELM. When the same number of training samples is selected, the pooling strategy has the highest recognition rate. In practical applications, the selection of appropriate pooling methods according to the actual situation of data is conducive to achieving better application results. Due to the uniting of CNN and RELM, CNN-RELM have the advantages of CNN and RELM and it is easier to learn and faster in testing. The future work includes improve the generalized ability and further reduce the training time.

Acknowledgement

This work is supported by the National Natural Science Foundation of China (Grant No. 61402227), the Natural Science Foundation of Hunan Province (No.2019JJ50618) and the project of Xiangtan University (Grant No. 11kz/kz08055). This work is also supported by the key discipline of computer science and technology in Hunan province, China.

Conflict of interest

The authors declare there is no conflict of interest.

References

1. Y. Bengio, Learning deep architectures for AI, *Found. Trends Mach. Learn.*, **1** (2009), 1–71.
2. F. Schroff, D. Kalenichenko and J. Philbin, *FaceNet: A Unified Embedding for Face Recognition and Clustering*, 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), United States of America, 2015, 815–823. Available from: https://arxiv.org/pdf/1503.03832.pdf.
3. G. Huang, Z. Liu, L. van der Maaten, et al., *Densely Connected Convolutional Networks,* 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), United States of America, 2017, 4700–4708. Available from: https://arxiv.org/abs/1608.06993.
4. Z. Q. Zhao, P. Zheng, S. T. Xu, et al., Object Detection with Deep Learning: A Review, *IEEE Trans. Neural Networks Learn. Syst.*, (2019), 1–21.
5. Y. L. Cun, L. D. Jackel, B. Boser, et al., Handwritten digit recognition: Applications of neural network chips and automatic learning, *IEEE Commun. Mag.*, **11** (1989), 41–46.
6. I. J. Goodfellow, J. P. Abadie, M. Mirza, et al., *Generative Adversarial Networks,* Advances in neural information processing systems, 2014, 2672–2680. Available from: https://arxiv.org/abs/1406.2661.
7. G. E. Hinton, S. Osindero and Y. W. Teh, A fast learning algorithm for deep belief nets, *Neural Comput.*, **18** (2006), 1527–1554.
8. Y. Z. Xu, X. J. Yao, X. Li, et al, Target detection in high resolution remote sensing images based on full convolution network, *Bull. Surv. Mapp.*, **1** (2018), 77–82.
9. M. S. Hasan, *An application of pre-trained CNN for image classification*, 2017 20th International Conference of Computer and Information Technology (ICCIT), 2017. Available from: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8281779.

10. X. Qin, Y. Zhou, Z. He, et al., *A Faster R-CNN Based Method for Comic Characters Face Detection*, 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), 2017, 1074–1080. Available from: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8270109.
11. R. Ranjan, V. M. Patel and R. Chellappa, HyperFace: A Deep Multi-Task Learning Framework for Face Detection, Landmark Localization, Pose Estimation, and Gender Recognition, *IEEE Trans. Pattern Anal. Mach. Intell.*, **41** (2019), 121–135.
12. S. Xie and H. Hu, Facial Expression Recognition Using Hierarchical Features with Deep Comprehensive Multipatches Aggregation Convolutional Neural Networks, *IEEE Trans. Multimedia*, **21** (2019), 211–220.
13. G. Chen, C. Li, W. Wei, et al., Fully convolutional neural network with augmented atrous spatial pyramid pool and fully connected fusion path for high resolution remote sensing image segmentation, *Appl. Sci.*, **9** (2019), 1816.
14. A. Kulikajevas, R. Maskeliūnas, R. Damaševičius, et al., Reconstruction of 3D Object Shape Using Hybrid Modular Neural Network Architecture Trained on 3D Models from ShapeNetCore Dataset, *Sensors*, **19** (2019), 1553.
15. G. B. Huang, Q. Y. Zhu and C. K. Siew, Extreme learning machine: A new learning scheme of feedforward neural networks, *Neural Networks*, **2** (2004), 985–990.
16. C. He, F. Xu, Y. Liu, et al., A fast kernel extreme learning machine based on conjugate gradient, *Network Comput. Neural Syst.*, **29** (2018), 70–80.
17. G. B. Huang, X. Ding and H. Zhou, Optimization method based extreme learning machine for classification, *Neurocomputing*, **74** (2010), 155–163.
18. W. Zou, F. Yao, B. Zhang, et al., Back Propagation Convex Extreme Learning Machine, *Proc. ELM*, **9** (2016), 259–272.
19. C. He, Y. Liu, T. Yao, et al., A fast learning algorithm based on extreme learning machine for regular fuzzy neural network, *J. Intell. Fuzzy Syst.*, **36** (2019), 3263–3269.
20. W. Deng, Q. Zheng and L. Chen, *Regularized Extreme Learning Machine*, 2009 IEEE Symposium on Computational Intelligence and Data Mining, 2009, 389–395. Available from: https://ieeexplore.ieee.org/abstract/document/4938676.
21. W. Y. Deng, Q. H. Zheng, L. Chen, et al., Study on extreme learning method of neural network, *Chin. J. Comput.*, **33** (2010), 279–287.

AIMS Press