An Optimized Seven-Layer Convolutional Neural Network with Data Augmentation for Classification of Chinese Fingerspelling Sign Language

Yalan Gao, Rongxin Zhu, Ruina Gao, Yuxiang Weng, and Xianwei Jiang(✉)

Nanjing Normal University of Special Education, Nanjing 210038, China
jxw@njts.edu.cn

Abstract. Sign language recognition especially finger language recognition facilitates the life of deaf people in China. It overcomes many difficulties and provides convenience for deaf people's life. In this paper, we used the advanced convolutional neural network to extract the different characteristics of the input. We created an optimized seven-layer CNN, including five convolution layers for feature extraction and two fully connected layers for classification to enhance the original signal function and reduce noise after operation. Some advanced techniques such as batch normalization, ReLu and dropout were employed to optimize the neural network. Meanwhile, we adopted data augmentation technology, which not only expanded the data set and improve the performance of machine learning algorithm, but also avoided the over-fitting problem. The experimental results show that the average recognition accuracy reaches 91.99 ± 1.21%, which indicate an excellent property.

Keywords: Convolutional neural network · Data augmentation · Chinese fingerspelling sing language · Batch normalization · ReLU · Maximum pooling · Dropout

1 Introduction

Sign language is a language for people with hearing and speech impairments to communicate with each other. People with hearing disabilities often combine gestures, body movements, and facial expressions to express themselves. Due to the convenience of fingerspelling sign language recognition, fingerspelling sign language recognition has attracted more and more attention and research. Fingerspelling language, an abbreviation for finger language, was developed specifically for the deaf and mute in China. It is a symbolic spelling of letters in which the syllables of words are typed in the order of the Pinyin Chinese system. It is worth mentioning that China is a vast country, and the same meaning is often confused with different sign language meanings due to different expressions in different regions. For this reason, a sign language containing only 30 letters is more accurate [1] and easier to recognize when facial expressions are not included. Images of letters for Chinese fingerspelling sign language are shown in Fig. 1.

W. Fu et al. (Eds.): ICMTEL 2021, LNICST 388, pp. 21–42, 2021.
https://doi.org/10.1007/978-3-030-82565-2_3

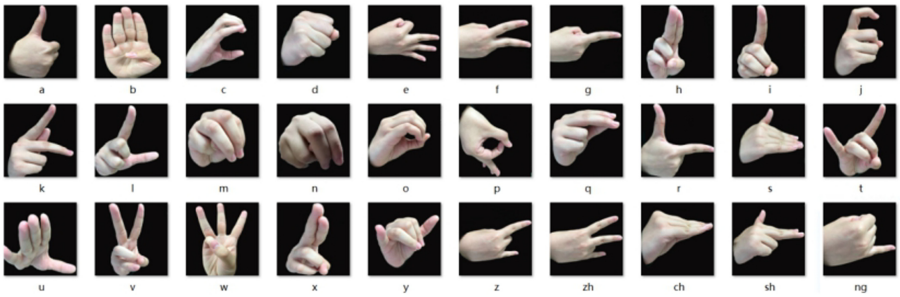

Fig. 1. Alphabets in sign language

Expression is the instinct and nature of human beings, and communication is equally important for deaf people. Like our Chinese language, sign language is the mother tongue of the deaf, and its use is becoming more and more popular nowadays. However, it is not easy for non-professionals to learn sign language, so it is difficult for people who have not mastered sign language to communicate with deaf and dumb people. We can choose to communicate with deaf people by means of sign language interpreters. However, this method is extremely inconvenient and expensive. In order to enable deaf people to communicate with normal people, we need a cheaper and more convenient solution. For a long time, scientists have been trying to find a way to make it easier for deaf and normal people to communicate with each other. A turning point in this field is the finger language recognition system. This system is designed to recognize the gestures of a deaf person and translate them into the local language using text or speech. This technology has had a positive effect on the social status of deaf people, allowing them to better adapt to society and enjoy normal civil rights to a greater extent than the average person.

Finger language recognition can be divided into wearable device recognition, touch technology recognition and computer vision recognition. The development of finger language recognition technology dates back to 1993, and gesture recognition technology has been adapted to speech and character recognition technology, Darrell and Pentland applied the dynamic time warp (DTW) of speech recognition to dynamic gesture recognition [2]; Rung-hui Liang et al. used gloves to collect raw data on 51 basic positions, including six directions and 8 The average recognition rate of consecutive sentences composed of these gestures was 80.4% [3]; Jiangqin Wu et al. combined a neural network with a learning decision tree to develop a recognition model and constructed a Chinese finger spelling gesture recognition system using a data glove [4]; later, Rung-hui Liang et al. used gloves to collect raw data for 51 basic positions, including six directions and eight movements, and then they simulated a dictionary system capable of recognizing 250 Taiwanese symbols using hidden Markov models (HMMs). The average recognition rate of consecutive sentences composed of these gestures was 80.4% [5]; Chuanbo Weng et al. used Bayesian gesture segmentation to model skin color and then combined it with skin color, Motion, and gesture recognition of contour shape information greatly improved segmentation accuracy [6]. The DWT algorithm builds a reference model by extracting the signal feature parameters of the reference model and storing them in the database of the reference model, builds a test model by extracting the signal feature

parameters of the gestures to be recognized, and then adds the total distance between the frame vectors of the reference model and the test model to obtain the computational result showing that the shorter the total distance, the higher the similarity. Finally, the smallest distance is selected as the matching result. Although this recognition algorithm is relatively simple and efficient, it is computationally intensive, takes up a large memory space, and has a long response time. The hidden Markov model-based approach is superior to time series modeling, but it requires independent sequences of gesture movements, which are often interdependent. In addition, once the models of these traditional mechanical learning methods are trained during the training phase, they do not change throughout the recognition process.

We proposed this algorithm based on the foundations of many predecessors. Ameen et al. propose a model for ASL letter recognition using CNNs [7]. Mohanty et al. proposed a different deep learning framework for recognizing static gestures with CNNs on complex backgrounds and under different lighting conditions, and obtained good recognition results [8]. The advanced deep learning techniques and advances in convolutional neural networks (CNNs), in particular, CNNs, which have completely surpassed traditional gesture recognition methods. They can achieve maximum performance without the need to manually design features.

The purpose of this article is to implement an optimized 7-layer convolutional neural network and add data augmentation to improve the accuracy of Chinese fingerspelling sign language recognition. We also improve the validity of the test data by combining pooling, batch normalization, and dropout techniques to overcome the inefficiency of pre-training and improve CNN accuracy and usability. The rest of this article is arranged as follows: Sect. 2 describes the data set, Sect. 3 specifically introduces the Chinese fingerspelling sign language recognition methods used in this article, Sect. 4 describes the experiment process, Sect. 5 provides discussions, and observations and acknowledgements are given in the last section.

2 Dataset

2.1 Data Collection and Image Preprocessing

We used a camera to shoot Chinese finger gesture images and established a related gesture data set, which contained 44 samples (each sample covers 26 basic letters and 4 tongue sounds, commonly used pronunciations, words, a total of 30 categories) 1,320 sheets 1080 × 1080 pixels photos. Then Photoshop CS was used to reduce noise and keep the hand-shaped area, then adjusted the size of the picture to 256 × 256, and finally saved it in tif format to ensure that basically no image information is lost. (See Fig. 2).

During the shooting, we fully considered the differences in sign language gestures used by individuals to make the captured images more convincing. (See Fig. 3).

2.2 Data Preprocessing

Through the above operations, the preprocessed images have the same size and background, and have no effects on the test results. At the same time, according to experimental requirements, we extract 80% of each type of sign language image as training samples, and the remaining images are used as experimental samples. (See Table 1).

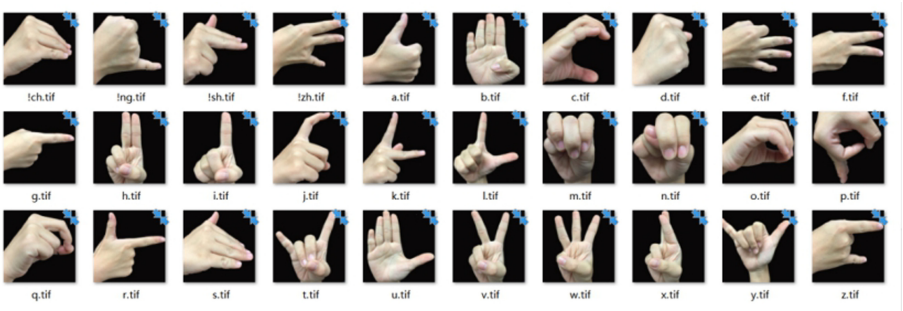

Fig. 2. Thirty classification source images of a sample

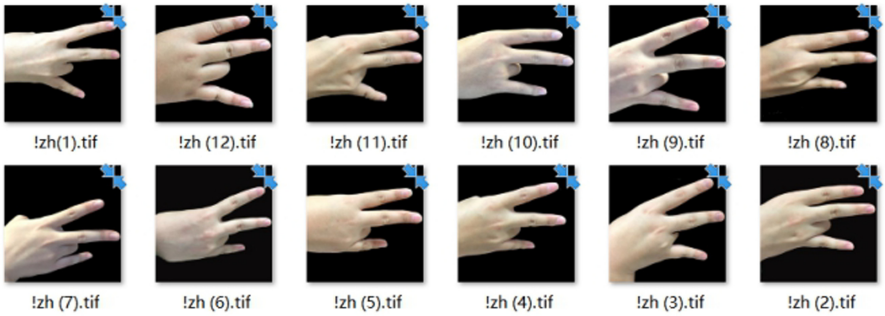

Fig. 3. Different people's sign language gestures about 'zh'

Table 1. Distribution of data integration

Type of data	Basic letters	Retroflex consonant	Total
Training set	915	140	1055
Test set	229	36	265
Total	1144	176	1320

3 Methodology

In this algorithm, the data augmentation technology is firstly used to increase the data scale and improve the reliability and accuracy of the experimental data. We mainly use convolutional neural network for image processing, including 5 convolutional layers for function extraction and 2 full connection layers to extract different features and reduce noise after operation. The use of batch normalization, ReLU, dropout and other advanced technologies to optimize the neural network not only provides convenience for our calculation but also improves the performance of the convolutional neural network.

3.1 Convolutional Layer

Convolutional neural network (CNN) is a kind of feedforward neural network with deep structure, which contains convolution computation and has the ability of representational learning. It can carry out large-scale image processing and is often used to analyze visual images. The convolutional layer is a part of the convolutional neural network, and it is the core of the convolutional neural network.

In a convolutional neural network, each convolutional layer is composed of several convolution units. The purpose of convolution operation is to extract different features of the input. The first layer of convolution may only be able to extract the edge of low-level features, while through multi-layer convolution, more complex features can be iteratively extracted from low-level features [9].

An important feature of convolution operation is that the original signal features are enhanced and noise is reduced after operation. The convolution algorithm is as follows:

Assuming that the size of the input image is 5×5 (a grid and a pixel), we had a convolutional neuron. The sliding window goes 2×2 and the step size is 2. When sliding the window, we found that one pixel cannot be obtained, so we added another layer of filling value to obtain all the pixels.

Our convolutional neuron started with a 3×3 matrix, and then selected a 3×3 matrix in the image to perform inner product calculation with the convolutional neuron. The result was obtained by adding 1 after the convolution operation [10, 11]. (see Fig. 4).

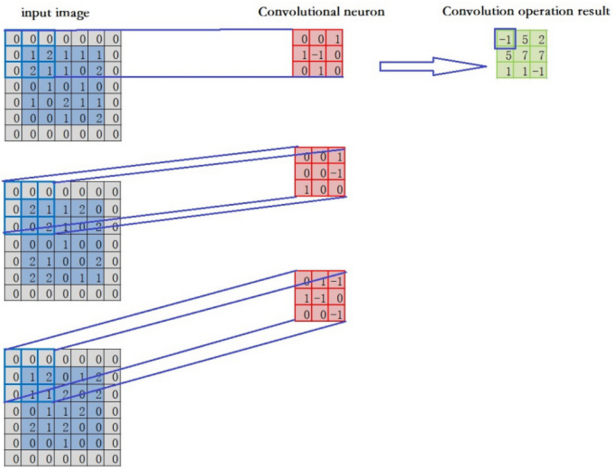

Fig. 4. The process of convolutional neuron network

The derivation formula of convolution layer output is as follows:

$$O = \frac{P - H + 2E}{S} + 1 \tag{1}$$

Where, the size of the input image is $P \times P$, the size of the convolution kernel is $H \times H$, the step size is S, the filling pixel is E. From the formula, we can also deduce that the size of our output feature graph is 3×3, that is, the green matrix in Fig. 4.

3.2 ReLU Function

ReLU

Rectified Linear Unit (ReLU), also known as modified linear element, is essentially a linear function and is a commonly used activation function in artificial neural networks. Its expression is as follows:

$$ReLU(x) = \begin{cases} 0, x \leq 0 \\ x, x > 0 \end{cases} \tag{2}$$

The image of the ReLU function is shown in Fig. 5:

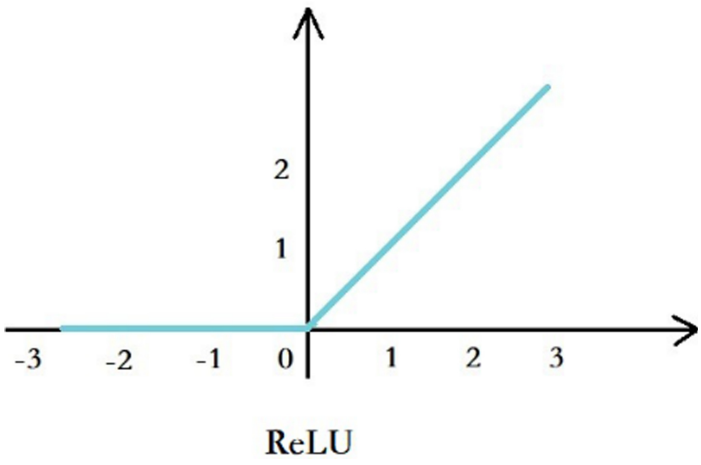

Fig. 5. The graphical representation of the ReLU function

As can be seen from the figure, when the input value is negative, the output value is 0, while the input value is integer, the output value is A. We can understand from the perspective of the neurons, neurons in negative cases will not be activated, only under the condition of positive neurons can be activated, this will reduce the network density, thus simplifies the calculation process, and improve calculation efficiency and better training data fitting, in CNN, when the model increased the n layer, theoretically ReLU neuron activation rate would be reduced [12].

The ReLU function uses bionics to debug the activity of neurons, using linear correction and regularization. Typically, about 50 percent of neurons in a neural network using modified linear units are active. Moreover, the ReLU function avoids the problem of gradient explosion and gradient disappearance and can stabilize the convergence rate.

LReLU

With Leaky Rectified Linear Unit (LReLU), when X is negative, there will be no output of all zero. The formula is as follows:

$$LReLu(x) = \begin{cases} x, & x \geq 0 \\ ax, & x < 0 \end{cases} (a\ is\ usually\ 0.01) \tag{3}$$

The function image is shown in Fig. 6.

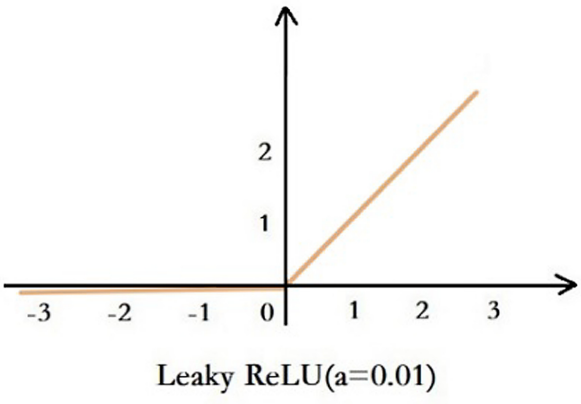

Fig. 6. The graphical representation of the LReLU function

ReLU is vulnerable in training, because it turns all input negative to 0. When the setting values are large, it will easily lead to the inactivation of neurons, and will not be activated in any form or way, resulting in the death of neurons. LReLU alleviates the problem of neuronal death. When the input is less than 0, it can maintain a certain output without causing permanent inactivation of neurons.

RReLU

Randomized Leaky Rectified Linear Unit (RReLU) functions with Leaky are an improvement on Leaky ReLU.

The formula is as follows:

$$RReLU(x) = \begin{cases} x, & x \geq 0 \\ ax, & x < 0, a \in [0, 1) \end{cases} \tag{4}$$

During model training, a is an arbitrary value, but during model testing, a becomes a fixed value. Compared with LReLU, the a in RReLU is a random variable derived from the probability model of continuous uniform distribution U (l, u). Theoretically, we could get the function image in Fig. 7:

Since a generates random allocation randomly in the U probability model, RReLU is more effective in the confrontation with overfitting.

3.3 Pooling Layer

Pooling layer is the network layer structure of convolutional neural network different from traditional neural network. Pooling layer is generally located behind the convolutional layer, as shown in Fig. 8.

The network model structure using pooling layer can speed up computer calculation and effectively prevent the occurrence of overfitting problem. Currently, almost all the mainstream convolutional neural network models include the pooling layer [13].

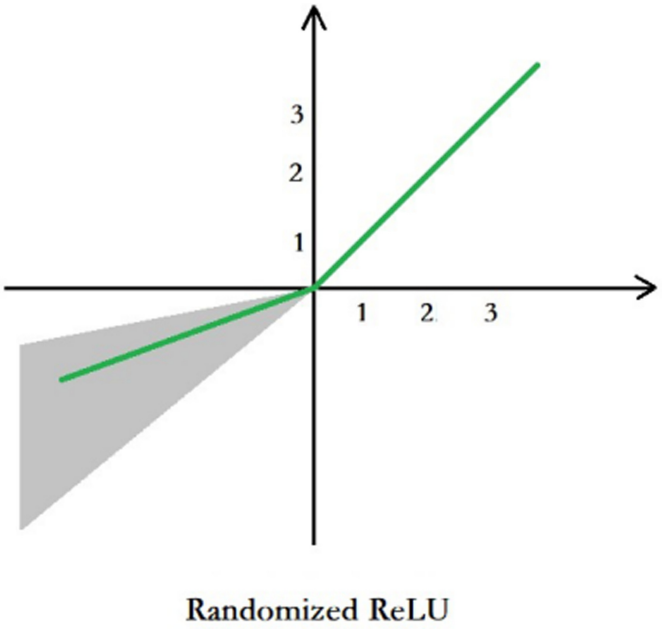

Fig. 7. The graphical representation of the RReLU function

Fig. 8. Pooling layer connection location

Giving a feature mapping group $X \in R^{A \times B}$ and input it into the pooling layer 1^P. Here, the feature map X^d in this group can be subdivided into several sub-regions $R^d_{\alpha,\beta}$ and $1 \leq \alpha \leq A$, $1 \leq \beta \leq B$. In the algorithm of the seven-layer convolutional neural network in this paper, the convolution operation often appeared the problem of over-fitting and heavy computation. In order to solve these problems, a pooling operation was added after the convolution operation to reduce the number of parameters corresponding to the features [14]. In the pooling layer, the calculation method is not the weighted sum of the corresponding nodes, but the simpler operation of maximum or average value. The pooling layer that uses the maximum value operation is called the Max pooling layer, and the pooling layer that uses the average value operation is called the Average pooling layer. The maximum pooling has a good inhibitory effect on the estimated mean deviation caused by parameter errors of the convolutional layer, and the average pooling can reduce the estimated error caused by the limited size of adjacent areas. Currently, the maximum pooling layer is most commonly used [15]. Figure 9 and Fig. 10 respectively shows the calculation methods of maximum pooling and average pooling.

Two common methods of pooling are as follows:

Max pooling refers to taking out the max value of all neurons in the corresponding area, which can be expressed as:

$$Y^d_{\propto,\beta} = \max(X_i),\ \ i \in R^d_{\propto,\beta} \tag{5}$$

In the above formula, X_i is the activation value of each neuron in region $R^d_{\propto,\beta}$.

Average pooling refers to the calculation of the average value of all neurons in the region as output, which is expressed as:

$$Y^d_{\propto,\beta} = \frac{1}{R^d_{\propto,\beta}} \sum\nolimits_{i \in R^d_{\propto,\beta}} X_i \tag{6}$$

Calculation method of maximum pooling and average pooling:

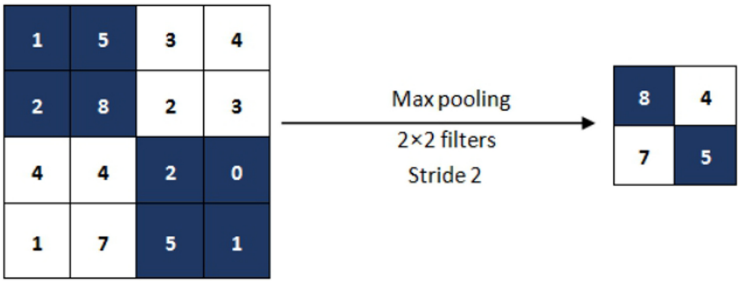

Fig. 9. Max pooling operation

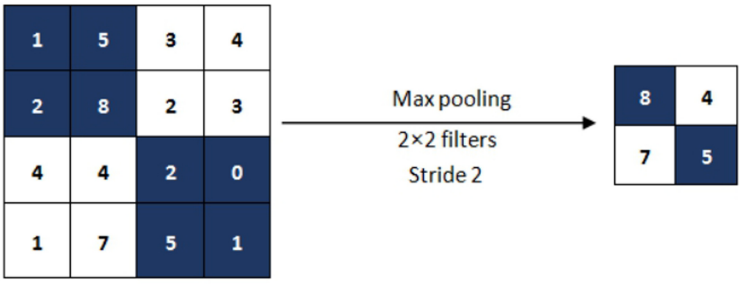

Fig. 10. Average pooling operation

Pooling can not only bring a larger view of the network architecture, but also maintain the invariability of some small local morphological changes in the image, and effectively reduce the number of neurons [16].

3.4 Batch Normalization

Batch normalization (BN) is often used to optimize the deep neural network. The method cannot merely improve the training speed, but also relax the requirement of parameter

adjustment to some extent. In addition, it provides a regularization effect similar to dropout, preventing model overfitting [15]. The operation process of BN is as follows:

Batch mean (u is the batch size):

$$\mu_B = \frac{1}{\mu}\sum_{i=1}^{\mu} X_i \tag{7}$$

Batch variance:

$$\sigma_{B^2} = \frac{1}{\mu}\sum_{i=1}^{u} (X_i - \mu_B)^2 \tag{8}$$

The normalized:

$$\widehat{x_i} = \frac{x_i - \mu}{\sqrt{\sigma^2 + \gamma}} \tag{9}$$

Scaling and shifting:

$$Y_i \leftarrow \gamma\widehat{x_i} + \beta = BN_{\gamma,\beta}(X_i) \tag{10}$$

Where, μ_B stands for batch mean, u stands for batch size, σ_{B^2} stands for batch variance, and $\widehat{X_i}$ stands for normalized operation.

BN handles each data and follows the normal distribution of N (0, 1), reducing the changes in the distribution of internal neurons. The structure of BN is shown in Fig. 11.

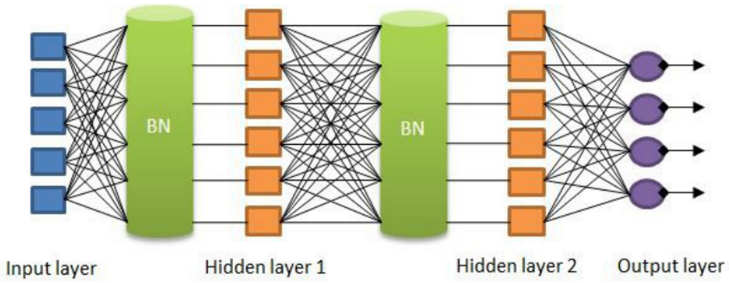

Fig. 11. Batch normalization

The deep learning network model can perform target recognition well. It is based on the assumption that the training set data and the test set data are independently and equally distributed. In the traditional training method, the data distribution of each batch is different, so the training is difficult, and the model is usually converged by reducing the learning rate. The deep learning network model is constantly changing, so the input value of each layer will have some deviation, which will affect the test accuracy [17]. In the use of BN, a relatively high learning rate can be selected to accelerate model convergence, improve training efficiency and enhance usability. Meanwhile, it can also standardize the process, thus eliminating dropout, simplifying network structure and improving utilization efficiency.

3.5 Dropout

Over-fitting problems often occur when training neural networks, which are specifically reflected in the following aspects: the loss function on the test data is relatively large, the prediction accuracy is low, and on the contrary, the prediction accuracy is high. When encountering over-fitting problems, the obtained model is almost unusable, but dropout can effectively alleviate the over-fitting problem [17, 18].

The dropout was proposed by Hinton in 2012 and its principle is simple: in a training iteration, neurons in each layer (total X) are randomly eliminated with probability Y, and the remaining $(1 - Y) \times X$ neurons are used to train the data in this iteration. In general, the effect is best when $Y = 0.5$. Half of the neurons are discarded during training, and only half of the remaining neurons can be activated, and the randomly generated network structure is the most diverse.

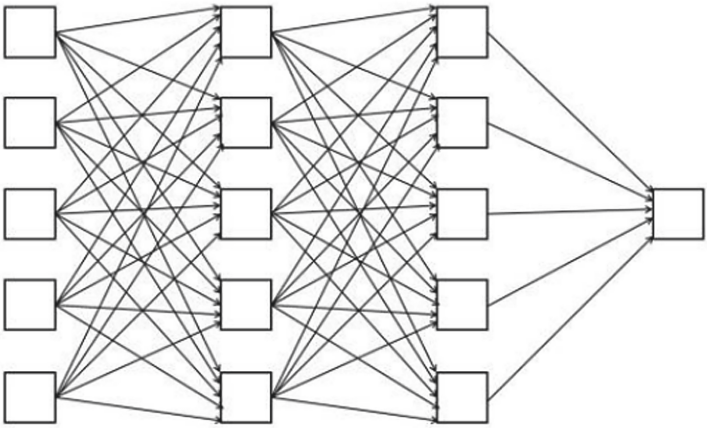

Fig. 12. The general neural network

As shown in the above figures, Fig. 12 is a general neural network, and Fig. 13 is a dropout neural network. We can conclude that the neural network is simplified after the application of this technology, easier to train, and can effectively prevent the occurrence of overfitting.

3.6 Fully Connected Layer

In the convolutional neural network, one or more fully connected layers are connected after multiple convolutional and pooling layers. Each node of the full connection layer is connected with all the nodes of the upper layer (as shown in Fig. 14), which is used to synthesize the features extracted from the front edge, so the weight parameters of the layer are the most. After the current convolutional layer has captured enough features to recognize the image, the next step is how to classify them. Fully connected layer in

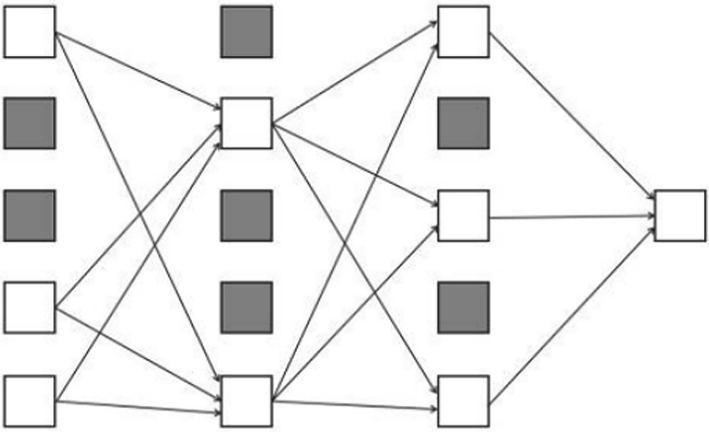

Fig. 13. The dropout neural network

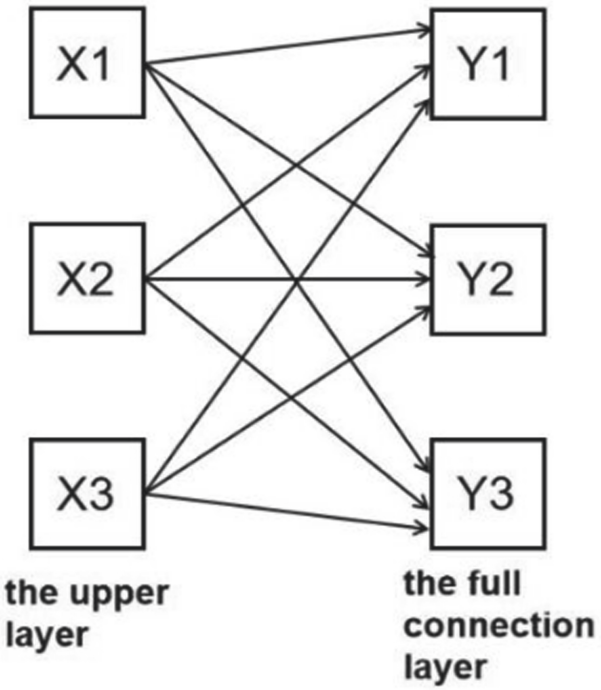

Fig. 14. The connection between the full connection layer and the upper layer

the whole convolution neural network is similar to "classifier", which is mainly used for classification, mapping distributed features to the sample label space.

Each output of the fully connected layer can be viewed as each node of the previous layer multiplied by a weight coefficient X and adding a bias value Y. For example, if

there are $30 \times 2 \times 2$ neuron nodes in the input and 400 nodes in the output, a total of $30 \times 2 \times 2 \times 400 = 48000$ weight parameters X and 400 bias parameters Y.

In order to improve the performance of the convolutional neural network, we added the ReLU function to the excitation function of each neuron in the fully connected layer. The output value of the last fully connected layer was passed to an output, which can be classified by softmax regression. This layer is called softmax layer.

3.7 Data Augmentation

Large amounts of data can improve the performance of machine learning algorithms and avoid overfitting problems. Xiang Yu et al. created a new data augmentation framework called SCDA (Scaling and Contrast limited adaptive histogram equalization Data Augmentation) for accurate classification of breast abnormalities based on ResNet-50 [19]. For detection of COVID-19, Motamed Saman et al. proposed a new GAN architecture by using the data augmentation technology for augmentation of chest X-rays for semi-supervised detection of pneumonia and COVID-19 using generative models [20]. Our data set is going to include the different conditions, such as different orientation, position, scale, brightness, and so on. However, in the actual collection of data, the number of data sets we collect is limited, that is to say, collecting large amount of sample data is a challenging task. By performing data augmentation, we can solve the problem of sample data and prevent neural networks from learning unrelated characteristics and fundamentally improve overall performance [21].

In a convolutional neural network, if it can accurately classify objects under different circumstances, it means that the neural network is stable. CNN has invariance to shift, viewpoint, size and lighting and this is essentially a prerequisite for data augmentation. Data augmentation is an effective technique to improve the accuracy of image classifiers and can also help CNN learn more powerful functions. In this article, we use the following six methods for data augmentation.

PCA Color Augmentation

PCA color augmentation is mainly used to adjust the brightness, saturation and contrast of the image. In order to maintain the validity of the artificial image, the first step is to calculate the main component analysis (PCA) of the training dataset to restore the color of its distribution spindle [22]. Then, the artificial images are created by constantly adjusting multiples of the principal components of the data set. Figure 15 shows the contrast before and after the PCA color augmentation.

Affine Transform

The affine transformation (AFT) is a clutter operation that randomly changes the pixel position of an image. The affine transformation of an image $f(p, q)$ of size $N \times N$ pixels is calculated by (p', q') function is represented as follows:

$$\begin{bmatrix} p' \\ q' \end{bmatrix} = AFT\{(p, q), N\} = \begin{bmatrix} x + hp \\ y = kq \end{bmatrix} (mod\ N) \tag{11}$$

Where "mod" represents the modal operation, x and y are two random numbers between 1 and N, and h, k is selected by their relative prime number relative to N. This

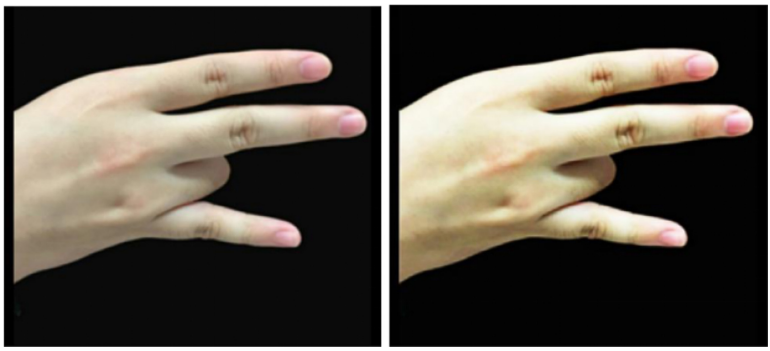

Fig. 15. The picture adds color augmentation before and after contrast

choice of h and k causes the AFT to map (p, q) to the unique pixels in the conversion coordinates. If h and k are not relative prime numbers, AFT maps different pixels to the same pixel in the transformed coordinates. After AFT, the total energy of the input image remains the same [23].

Noise Injection

The core of noise injection is to randomly disturb each pixel RGB of the image by adding the random value matrix sampled from the Gaussian distribution, so as to produce some new noise polluted images. At the same time, it can also help CNN learn more powerful functions [24].

Scaling

The scaling method includes scaling inward and outward. When zoomed out, the final image size is larger than the original image size. Similarity, when scaled inward, the final image size will be smaller than the image size. Most image frames cut out a section from a new image that is equal in size to the original image. The Fig. 16 shows the contrast of image zoomed in.

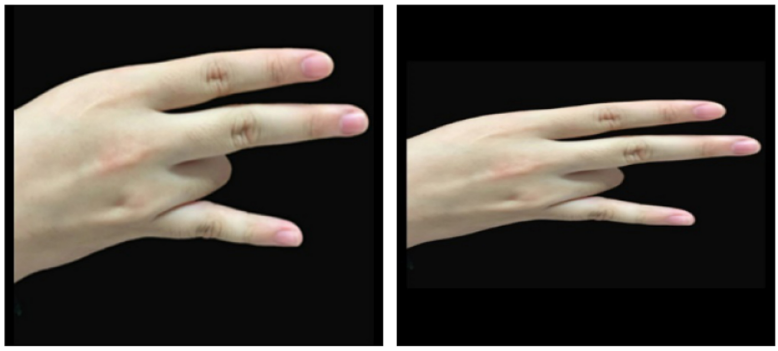

Fig. 16. The contrast of image zoomed in

Random Shift

Random shift only involves moving images in the X or Y direction (or both). In the following example, we assume that the image has a black background outside its boundaries and is shifted appropriately. This enhancement is useful because most objects can be located almost anywhere in the image. This forces your convolutional neural network to see all corners. The Fig. 17 shows the effect of rotating 90 degrees counterclockwise.

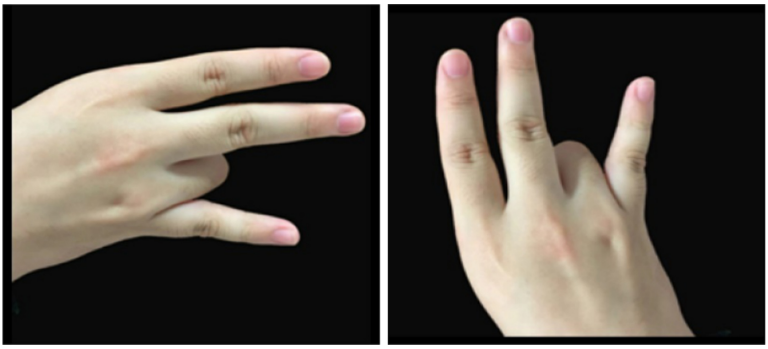

Fig. 17. The contrast of image shift

Gamma Correction

Gamma correction is used to correct the nonlinear photoelectric conversion characteristics of sensors in electronic devices such as cameras, which edits the gamma curve of an image to make nonlinear tonal edits to the image [25]. Gamma represents a diagonal line between the output value of the image and the input value and the gamma value is usually 2.3. In this paper, we produce a number of different brightness images by constantly adjusting the gamma value. The gamma correction curve is shown in Fig. 18.

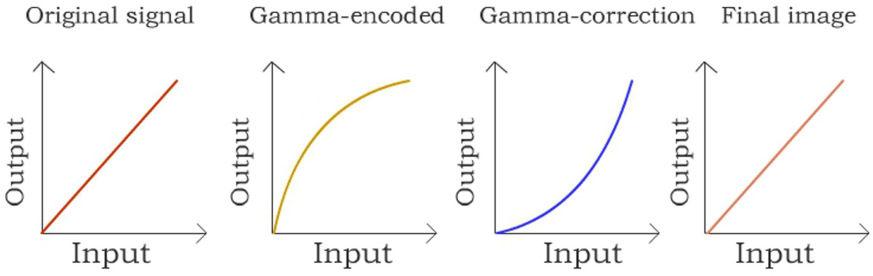

Fig. 18. The process of gamma correction of images

4 Experiment Results

4.1 Experiment Setting

Our experiment was run on the personal computer with windows 7 system, whose configuration is as follows: 3.2GHz Intel(R) Core (TM) i5 CPU, 16 GB RAM and NVIDA GeForce Graphics Processor. To overcome the issue of randomness, the training and test were implemented more times, and then the mean and standard deviation were obtained. The main parameters of training option are provided as follows: Adam (Adaptive momentum) algorithm is selected as training algorithm, MaxEpochs equals 30, InitialLearnRate is 0.01, MiniBatchSize is set to 256, LearnRateSchedule is piecewise, LearnRateDropFactor is set to 0.1 and L2Regularization is 0.005. Finally, average accuracy is adopted to evaluate the identification.

4.2 Structure of Proposed CNN

An optimized seven-layer CNN was created, including five convolutional layers for feature extraction and two fully-connected layers for classification. As shown in Table 2, the detailed parameters settings were given. Where, some advanced techniques i.e. batch normalization, ReLU and pooling were applied to improve the performance. Meanwhile, the dropout rate was set to 0.4. In addition, softmax function was employed.

Table 2. Details of each layer in our CNN

Index	Layer	Filter/pool size	Filters	Stride
	Input	$256 \times 256 \times 3$		
1	Conv_BN_ReLU_1 (Pool)	7/3	32	3
2	Conv_BN_ReLU_2 (Pool)	3/3	64	3
3	Conv_BN_ReLU_3 (Pool)	3/3	128	1
4	Conv_BN_ReLU_4 (Pool)	3/3	256	3
5	Conv_BN_ReLU_5 (Pool)	3/3	512	1
6	FCL_1(Dropout = 0.4)			
7	FCL_2(Softmax)			

4.3 Statistical Analysis

As shown in Table 3, this table provides 10 runs results of our method. The average accuracy of identification reaches $91.99 \pm 1.21\%$, which is bolded in the column. Meanwhile, the highest accuracy achieves 93.36%, and the accuracy values of nine runs exceed 90%. Therefore, it can be considered that our method owns satisfactory effectiveness and stability.

Table 3. Ten runs of our method

Run	Our method
1	91.41
2	90.23
3	91.41
4	92.58
5	89.84
6	92.58
7	93.36
8	92.97
9	92.58
10	92.97
Average	**91.99 ± 1.21**

5 Discussions

5.1 Comparison to Pooling Method

Two common pooling methods i.e. maximum pooling and average pooling were tested in the experiment. Meanwhile, the parameter settings were unchanged. As shown in Table 4, comparison of average pooling and maximum pooling are listed in 10 runs. For a vivid comparison, the results are represented in Fig. 19. It can be seen that maximum pooling is obviously better than average pooling in term of average accuracy. Maximum pooling achieves the highest accuracy of 93.36% while average pooling gains the value of 92.58% in the highest accuracy. Moreover, the accuracy value of maximum pooling in each run is significantly greater than or equal to that of average pooling.

5.2 Effect of Data Augmentation

To verity the effectiveness of data augmentation (DA) in training, the same experiments were executed between using DA and without DA. The effect of data augmentation is indicated in Fig. 20. Average accuracy of method with DA reaches 91.99 ± 1.21%, which can increase about 2.9% than the accuracy of method without DA. Obviously, the data augmentation technique can generate more images to expand the dataset, which provides sufficient data and enhances the performance of identification. Furthermore, DA technology also cuts issue of over-fitting.

5.3 Comparison to State-of-the-Art Approaches

Our proposed method "CNN7-DA" was compared with five state-of-the-art approaches i.e. GLCM-MGSVM [26], HMI-RBF-SVM [27], CNN6-LReLU [28], WE-kSVM [29]

Table 4. Comparison of average pooling and maximum pooling

Run	Average pooling	Maximum pooling
1	89.06	91.41
2	90.23	90.23
3	89.45	91.41
4	88.28	92.58
5	89.84	89.84
6	91.02	92.58
7	89.45	93.36
8	90.23	92.97
9	90.63	92.58
10	92.58	92.97
Average	90.08 ± 1.18	91.99 ± 1.21

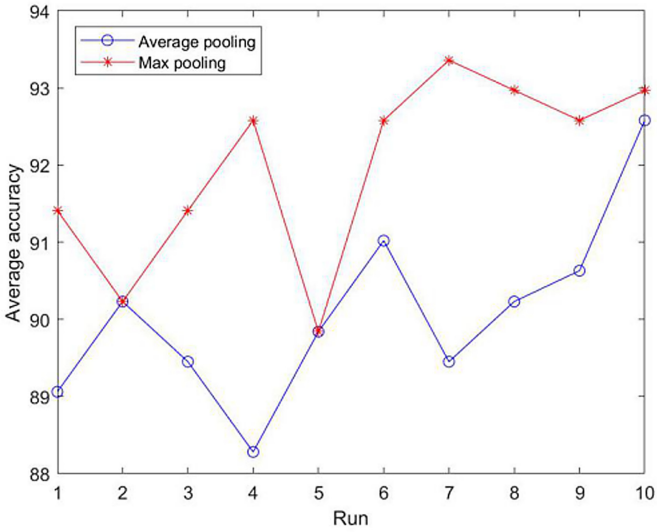

Fig. 19. Comparison of average pooling and maximum pooling

and AlexNet-TL [30] in the experiment. The average accuracy of GLCM-MGSVM, HMI-RBF-SVM, CNN6-LReLU, WE-kSVM and AlexNet-TL are 85.30%, 86.47%, 88.10%, 89.4% and 89.48%, respectively, which are denoted in Fig. 21. We can observe that proposed "CNN7-DA" obtains relatively better performance among six algorithms.

Two attributes are summarized to explain the reason why our method is superior. First and foremost, an optimized seven layers CNN was employed, which composed

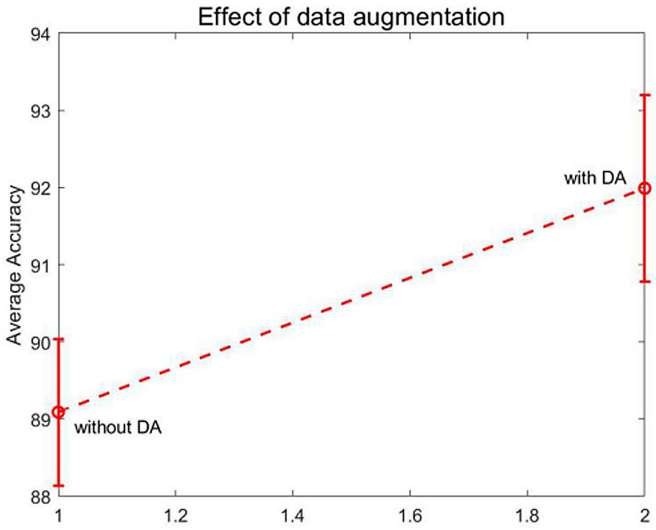

Fig. 20. Effect of data augmentation

BN, ReLU, pooling and dropout techniques into an advanced block. CNN can gain learnable weights through iteration and extract features automatically. Advanced block can overcome issue of over-fitting and improve performance. Secondly, application of DA extends image dataset and provides sufficiency of date, thus deep neural network generality can be enhanced.

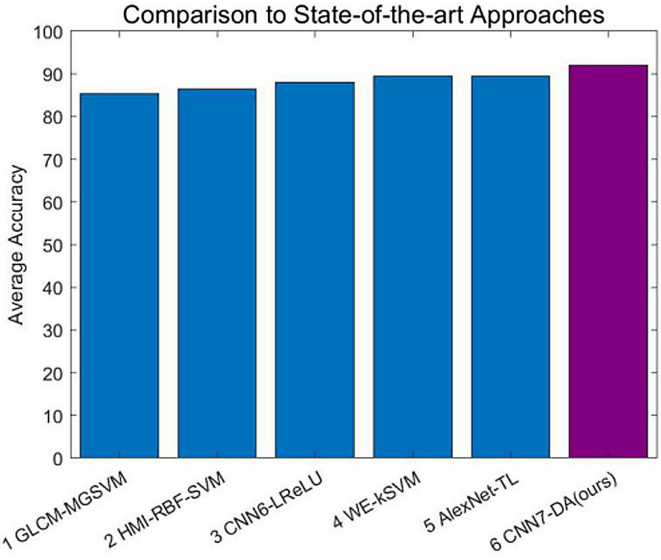

Fig. 21. Comparison to state-of-the-art approaches

6 Conclusions

In this study, a novel seven-layer CNN with advanced techniques block was proposed to identify Chinese fingerspelling sign language. Batch normalization and ReLU techniques are employed to accelerate convergence of learn and eliminate gradient disappearance. Pooling and dropout techniques are adapted to reduce dimensionality and overcome overfitting. Data augmentation technique provides sufficiency of image data. Our method achieves average accuracy of $91.99 \pm 1.21\%$, which is superior to five state-of-the-art approaches. The mainly explanation is indicated in Sect. 5.3 Comparison to state-of-the-art approaches.

In the future, we will try different levels of convolutional neural networks and try other training algorithms to further optimize parameters to improve overall performance. Meanwhile, we will try to verify our study in other application of recognition, for instance, Lip language recognition, movement recognition, rehabilitation gesture recognition, etc. are all potential areas.

Acknowledgements. This work was supported by National Philosophy and Social Sciences Foundation (20BTQ065), Natural Science Foundation of Jiangsu Higher Education Institutions of China (19KJA310002), The Philosophy and Social Science Research Foundation Project of Universities of Jiangsu Province (2017SJB0668).

References

1. Jiang, X., Satapathy, S.C., Yang, L., Wang, S.-H., Zhang, Y.-D.: A survey on artificial intelligence in Chinese sign language recognition. Arab. J. Sci. Eng. **45**(12), 9859–9894 (2020). https://doi.org/10.1007/s13369-020-04758-2
2. Premaratne, P.: Historical Development of Hand Gesture Recognition. In: Premaratne, P. (ed.) Human Computer Interaction Using Hand Gestures, pp. 5–29. Springer Singapore, Singapore (2014). https://doi.org/10.1007/978-981-4585-69-9_2
3. Liang, R.H., Ming, O.: A real-time continuous gesture recognition system for sign language. In: IEEE International Conference on Automatic Face & Gesture Recognition (1998)
4. Wu, J., Wen, G., Cheng, X.: A system recognizing Chinese finger-spelling alphabets based on data-glove input. Pattern Recogn. Artif. Intell. (1999)
5. Liang, R.-H.: A real-time continuous gesture recognition system for sign language. In: Proceedings of The Third IEEE International Conference on Automatic Face and Gesture Recognition (1998)
6. Weng, C., Li, Y., Zhang, M., Guo, K., Tang, X., Pan, Z.: Robust Hand Posture Recognition Integrating Multi-cue Hand Tracking. In: Zhang, X., Zhong, S., Pan, Z., Wong, K., Yun, R. (eds.) Edutainment 2010. LNCS, vol. 6249, pp. 497–508. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-14533-9_51
7. Ameen, S., Vadera, S.: A convolutional neural network to classify American sign language fingerspelling from depth and colour images. Expert Syst. **34**(3), e12197 (2017). https://doi.org/10.1111/exsy.12197
8. Mohanty, A., Rambhatla, S., Sahay, R.: Deep Gesture: Static Hand Gesture Recognition Using CNN. In: Raman, B., Kumar, S., Roy, P.P., Sen, D. (eds.) Proceedings of International Conference on Computer Vision and Image Processing, pp. 449–461. Springer Singapore, Singapore (2017). https://doi.org/10.1007/978-981-10-2107-7_41

9. Sun, J., He, X., Tan, W., Wu, X., Lu, H.: Recognition of crop seedling and weed recognition based on dilated convolution and global pooling in CNN. Trans. Chin. Soc. Agric. Eng. **34**(11), 159–165 (2018)
10. Yu, H., Ding, L., Shi, H., Hanchao, Y., Huang, T.S.: Computed tomography super-resolution using convolutional neural networks. In: IEEE International Conference on Image Processing (ICIP) (2017)
11. Wang, Shui-Hua., Hong, J., Yang, M.: Sensorineural hearing loss identification via nine-layer convolutional neural network with batch normalization and dropout. Multimedia Tools Appl. **79**(21–22), 15135–15150 (2018). https://doi.org/10.1007/s11042-018-6798-3
12. Banerjee, C., Mukherjee, T., Pasiliao, E.: An empirical study on generalizations of the ReLU activation function. In: The 2019 ACM Southeast Conference (2019)
13. Wang, Y., Liu, Z., Mu, X., Gao, S.: Modeling and verification of contact line transient temperature difference based on lifting or lowering the pantograph electric contacts. Chin. J. Sci. Instrum. **35**(12), 2663–2672 (2014)
14. Ying-bing, L.: Research on computer technology of remote supervisory and management system. In: Conference and Technology of West China (2010)
15. Li, D., Deng, L., Cai, Z.: Research on image classification method based on convolutional neural network. Neural Comput. Appl. **33**, 8157–8167 (2020). https://doi.org/10.1007/s00521-020-04930-7
16. Wang, S.-H., Lv, Y.-D., Sui, Y., Liu, S., Wang, S.-J., Zhang, Y.-D.: Alcoholism detection by data augmentation and convolutional neural network with stochastic pooling. J. Med. Syst. **42**(1), 1–11 (2017). https://doi.org/10.1007/s10916-017-0845-x
17. Wei, Z., Yang, J., Min, S.: A method of underwater acoustic signal classification based on deep neural network. In: 2018 5th International Conference on Information Science and Control Engineering (ICISCE) (2019)
18. Shen, X., Tian, X., Liu, T., Xu, F. Tao, D.: Continuous dropout. IEEE Trans. Neural Netw. Learn. Syst. 1–12 (2017)
19. Yu, X., Kang, C., Guttery, D., Kadry, S., Chen, Y., Zhang, Yu-Dong.: ResNet-SCDA-50 for breast abnormality classification. IEEE/ACM Trans. Comput. Biol. Bioinform. **18**(1), 94–102 (2021). https://doi.org/10.1109/TCBB.2020.2986544
20. Motamed, S., Rogalla, P., Khalvati, F.: Data Augmentation using Generative Adversarial Networks (GANs) for GAN-based Detection of Pneumonia and COVID-19 in Chest X-ray Images (2020)
21. Eckert, D., Vesal, S., Ritschl, L., Kappler, S., Maier, A.: Deep Learning-based Denoising of Mammographic Images using Physics-driven Data Augmentation. Presented at the (2020). https://doi.org/10.1007/978-3-658-29267-6_21
22. Vasconcelos, C.N., Vasconcelos, B.N.: Convolutional Neural Network Committees for Melanoma Classification with Classical and Expert Knowledge Based Image Transforms Data Augmentation (2017)
23. Singh, P., Yadav, A.K., Singh, K.: Color image encryption using affine transform in fractional Hartley domain. Opt. Appl. 47(3) (2017)
24. Igl, M., Ciosek, K., Li, Y., Tschiatschek, S., Hofmann, K.: Generalization in Reinforcement Learning with Selective Noise Injection and Information Bottleneck (2019)
25. Wang, Shui-Hua., et al.: Multiple sclerosis identification by 14-layer convolutional neural network with batch normalization, dropout, and stochastic pooling. Front. Neurosci. **12**, 818 (2018)
26. Jiang, X.: Isolated Chinese sign language recognition using gray-level co-occurrence matrix and parameter-optimized medium Gaussian support vector machine. Front. Intell. Comput.: Theory Appl. **1014**, 182–193 (2019)

27. Ya, G., et al.: Chinese Fingerspelling Recognition via Hu Moment Invariant and RBF Support Vector Machine. In: Zhang, Y.-D., Wang, S.-H., Liu, S. (eds.) ICMTEL 2020. LNICSSITE, vol. 327, pp. 382–392. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-51103-6_34
28. Jiang, X., Zhang, Y.-D.: Chinese sign language fingerspelling via six-layer convolutional neural network with leaky rectified linear units for therapy and rehabilitation. J. Med. Imag. Health Inform. **9**(9), 2031–2090 (2019)
29. Zhu, Z., Zhang, M., Jiang, X.: Fingerspelling identification for Chinese sign language via wavelet entropy and kernel support vector machine. Intell. Data Eng. Anal. **1177**, 539–549 (2020)
30. Jiang, X., Hu, B., Chandra Satapathy, S., Wang, S.-H., Zhang, Y.-D.: Fingerspelling identification for Chinese sign language via AlexNet-based transfer learning and Adam optimizer. Sci. Program. 2020, 1–10 (2020)