Research Article

Fingerspelling Identification for Chinese Sign Language via AlexNet-Based Transfer Learning and Adam Optimizer

Xianwei Jiang,^{1,2} Bo Hu,^{1,2} Suresh Chandra Satapathy,³ Shui-Hua Wang^{ID},^{4,5,6}
and Yu-Dong Zhang^{ID}^{7,8,9}

¹Nanjing Normal University of Special Education, Nanjing 210038, China

²Joint Accessibility Key Laboratory, China Disabled Persons' Federation, Nanjing 210038, China

³School of Computer Engineering, KIIT Deemed to University, Bhubaneswar, India

⁴School of Computer Science and Technology, Henan Polytechnic University, Jiaozuo, Henan 454000, China

⁵School of Mathematics and Actuarial Science, University of Leicester, Leicester LE1 7RH, UK

⁶School of Architecture Building and Civil Engineering, Loughborough University, Loughborough LE11 3TU, UK

⁷School of Informatics, University of Leicester, Leicester LE1 7RH, UK

⁸Guangxi Key Laboratory of Trusted Software, Guilin University of Electronic Technology, Guilin 541004, China

⁹Department of Information Systems, Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah 21589, Saudi Arabia

Correspondence should be addressed to Shui-Hua Wang; shuihuawang@ieee.org and Yu-Dong Zhang; yudongzhang@ieee.org

Received 12 February 2020; Accepted 28 March 2020; Published 19 May 2020

Academic Editor: Chenxi Huang

Copyright © 2020 Xianwei Jiang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

As an important component of universal sign language and the basis of other sign language learning, finger sign language is of great significance. This paper proposed a novel fingerspelling identification method for Chinese Sign Language via AlexNet-based transfer learning and Adam optimizer, which tested four different configurations of transfer learning. Besides, in the experiment, Adam algorithm was compared with stochastic gradient descent with momentum (SGDM) and root mean square propagation (RMSProp) algorithms, and comparison of using data augmentation (DA) against not using DA was executed to pursue higher performance. Finally, the best accuracy of 91.48% and average accuracy of $89.48 \pm 1.16\%$ were yielded by configuration M1 (replacing the last FCL8) with Adam algorithm and using 181x DA, which indicates that our method can identify Chinese finger sign language effectively and stably. Meanwhile, the proposed method is superior to other five state-of-the-art approaches.

1. Introduction

Nowadays, hearing-impaired people account for a large number, especially in China, where there are about 27.9 million deaf people [1]. Their lives, learning, and communication have all encountered unprecedented challenges. Sign language (SL) is their way of expression. But sign language contains a series of content and elements such as hand shape, movement, posture, and emotion, which is a relatively complicated system and not easy to be learned and mastered. Sign language translation and sign language recognition are two important solutions for above challenges. However, the former needs to arrange staff in

advance and is expensive, and the latter is receiving more and more attention due to the rapid development of intelligent technology.

Chinese Sign Language (CSL) is designed specifically for Chinese deaf and hearing-impaired groups [2]. It has distinct characteristics, that is, rich in semantics, wide in area, and various in expressions. As an important part of it, Chinese finger sign language is of great significance. First of all, it is simple and easy to learn, which consists of 26 monosyllabic letters (A to Z) and 4 double syllable letters (ZH, CH, SH, NG). The content is clear, unique, and easy to remember. Secondly, the pinyin function is mapped, which can be used as the basis for gesture sign language learning.

The fingerspelling in Chinese Sign Language is different from the letter expression of American Sign Language (ASL). The fingerspelling of American Sign Language is conducive to the direct expression of words or meanings, while the Chinese finger sign language represents the composition of a pinyin element in most of the time. Several finger sign languages constitute a complete meaning, which is not easy to produce ambiguity. In addition, Chinese finger sign language has advantages in representing abstract concepts and terminology.

Sign language recognition (SLR) refers to the utilization of computer technology to translate sign language into texts, images, audio, video, and natural language that can be understood and accepted. Generally, it can be divided into two types: one is based on sensors and another is based on computer vision. Especially, the recognition technology based on computer vision [3–6] is a popular trend, which is flexible to operate, easy to implement, low in cost, and reliable in technology. Many researchers have contributed to these areas by providing superior classification and recognition algorithms and their variants. Using four different support vector machine kernels, a fingerspelling recognition system focusing on Thai fingerspelling sign language achieved the average accuracies of 91.20%, 86.40%, 80.00%, and 54.67%, respectively [7]. But it only had 375 character pictures totally. Based on HMM, K-means, and ant colony algorithm, Li et al. [8] recognized Taiwan Sign Language with accuracy of 91.3%. However, their database just included 11 special Taiwan Sign Language words. The dynamic time warping (DTW) was presented by Lichtenauer et al. [9] and gained a recognition rate of 92.3%. They evaluated sign classification on a set of Dutch Sign Language with 120 diverse signs. In addition, the hierarchical conditional random field (HCRF) method was proposed by Yang and Lee [10]. Rao et al. [11] trained the ANN classifier to match words. Kumar et al. [12] used the hidden Markov model (HMM) method. Lee et al. [13] combined SVM with HMM.

Some other research studies of sign language recognition utilizing deep learning were introduced in [14–17]. Jiang [18] employed the Gray-Level Co-occurrence Matrix (GLCM) and Parameter-Optimized Medium Gaussian Support Vector Machine (MG SVM) method to identify isolated Chinese Sign Language. A 6-layer convolutional neural network with the leaky rectified linear unit (LReLU) technique for Chinese Sign Language fingerspelling recognition was proposed by Jiang [19].

Although these techniques and methods have achieved favorable results, they still have their own problems and shortcomings. For instance, HMM and DTW need to establish an effective template first and are not suitable for real-time system. Feature extraction is quite important for SVM, which will consume a huge amount of resources. The accuracy of HCRF can get great improvement potential. In contrast, with strong self-learning ability and organizational capability, neural networks bring a new dawn. In particular, transfer learning is fueling those mature and efficient neural networks.

In this paper, we proposed a AlexNet-based transfer learning method equipped with data augmentation and

Adam optimizer for fingerspelling identification of Chinese Sign Language and gained a stable average accuracy of $89.48 \pm 1.16\%$, which can be considered effective and significant.

Our contributions are as follows. (i) AlexNet-based transfer learning was introduced, which included some advanced techniques such as rectified linear unit function (ReLU), local response normalization (LRN), and dropout. (ii) Optimization algorithm Adam was utilized to accelerate the learning speed and enhance the performance. (iii) Data augmentation (DA) was used to provide sufficient training dataset, which strongly alleviated overfitting and raised the accuracy. (iv) Our study offered a new approach to smooth the barriers of communication between hearing-impaired people and healthy people.

2. Dataset

A self-built dataset of Chinese finger sign language based on computer vision was established, which was consisted of 1,320 images. The dataset is available upon e-mail request. According to the universal sign language standard issued by the state, Chinese finger sign language contains totally 30 sign language letters, 26 of which are single letters (A to Z) and 4 are double syllable letters (ZH, CH, SH, and NG). Figure 1 demonstrates thirty categories of Chinese finger sign language intercepted from sample images. These samples were preprocessed and normalized to 256×256 background-optimized images. Our experiment was executed with this private dataset including 1,320 images. Among them, 1050 images were used for training, and the rest were used for testing. The fingerspelling data used to support the findings of this study are available from the corresponding author upon request.

3. Methodology

3.1. Transfer Learning. An important reason why transfer learning catches the focus is that traditional machine learning applications require a large amount of labeled data, but these datasets may have problems such as distribution differences and training data expiration [20]. Fortunately, transfer learning can solve these issues. It can make full use of the previously labeled data and can guarantee the accuracy of the model on new tasks, which makes it get more and more attention.

Transfer learning is defined as the ability of a system to recognize and apply knowledge and skills learned in previous domains/tasks to novel domains/tasks. Transfer learning can be divided into instance-based transfer, feature-based transfer, and shared parameter-based transfer. The key to transfer learning is to find out which shared knowledge can be moved between different domains, design appropriate algorithms to extract and transfer common knowledge, and avoid negative transfer [21]. There are two general strategies for the application of transfer learning in deep learning. One is fine-tuning, which involves using a pre-trained network on the base dataset and training all layers in the target dataset; the other is freezing and replacing, for

FIGURE 1: Thirty categories of Chinese finger sign language.

instance, freezing all layers (the weights are not updated) except the last layer and training the last layer or freezing N layers before (that is, the freezing layer was selected by custom) and training the last few layers remaining. Figure 2 shows the application of fine-tuning idea of the transfer learning. It can be seen that mapping from a big dataset to a target small data can be quickly achieved through reasonable fine-tuning, and a new suitable model is established from a pretrained model [22].

In the image processing field, pretraining ImageNet is frequently chosen and its model is initialized. In this study, we initially trained a pretrained model on a subset of ImageNet having 1,000 categories and then transferred the previously learned knowledge to 30 categories of Chinese Sign Language recognition based on a small amount of private data, which is a comparatively simplified task. The entire structure of the neural network was retrained. Thus, the AlexNet-based transfer learning brought advantages for fingerspelling identification of Chinese Sign Language.

3.2. AlexNet: Structure and Layers. AlexNet has attracted much attention due to the fact that it was far ahead of the second winner in the 2012 ImageNet Large Scale Visual Recognition Challenge (ILSVRC-2012), which shocked the academic and industrial researchers at the time. It can be thought as a deeper and wider version of LeNet. LeNet contains the basic modules of convolutional neural networks, providing the basis for other deep learning models [23]. From a holistic perspective, AlexNet retains the original idea but introduces more tricks such as ReLU, LRN, and dropout. The rapid development of technologies such as the use of GPUs and resolution of data sample limits has provided an opportunity for AlexNet.

In terms of layers that associate with learnable weights, AlexNet consists of a total of eight layers, including five convolutional layers and three fully connected layers [24]. The structure of AlexNet is illustrated in Figure 3.

Although AlexNet is similar to traditional neural networks, it has its own characteristics. The main aspects on advanced techniques can be summarized as follows:

FIGURE 2: Fine-tuning idea of the transfer learning.

- (a) N learnable filters are contained in convolutional layers, and each filter can generate one feature map. Thus, finally a size of $N \times O_W \times O_H$ activation map will be obtained, which implemented the function of feature extraction. Suppose that we have a 3D input with size of $I_W \times I_H \times L$ and a filter with $F_W \times F_H \times L$; $O_W \times O_H$ indicates the size of output activation map. The value of O_W and O_H can be calculated in following formulas:

$$\begin{aligned} O_W &= 1 + \frac{(I_W - F_W + 2P)}{S}, \\ O_H &= 1 + \frac{(I_H - F_H + 2P)}{S}, \end{aligned} \quad (1)$$

where the number of filters is defined as N , S indicates the stride size, and P denotes padding size. The entire process of convolution is shown in Figure 4, which started from the input with a series of filters and finally output an activation map.

- (b) Replacing the traditional sigmoid function with rectified linear unit function (ReLU) is one of AlexNet's successful practical strategies. Because it is essentially a half-wave rectifier, it significantly speeds up the training process and prevents overfitting. In

FIGURE 3: Structure of AlexNet.

FIGURE 4: Illustration of convolution operation.

contrast, the sigmoid function is susceptible to gradient disappearance. It has been proven that the convergence speed of ReLU as an activation function in deep neural networks is about 7 times that of the traditional activation function. The formulas for both are expressed as follows:

$$\begin{aligned} \text{Sigmoid}(x) &= \frac{1}{1 + e^{-x}}, \\ \text{ReLU}(x) &= \max(0, x). \end{aligned} \quad (2)$$

- (c) AlexNet's another superior practical strategy is local response normalization (LRN) which was proposed by Krizhevsky et al. [25] to promote the convergence. Based on the computable neuron α_i with application of kernel i and nonlinear ReLU, the response-normalized neuron β_i can be computed as

$$\beta_i = \frac{\alpha_i}{\left(k + m \sum_{j=B}^E \alpha_j^2\right)^z}. \quad (3)$$

Among them,

$$\begin{aligned} B &= \max\left(0, i - \frac{n}{2}\right), \\ E &= \min\left(N - 1, i + \frac{n}{2}\right), \end{aligned} \quad (4)$$

where n denotes the number of contiguous kernel maps, which is the size of window channel, and N represents the total number of kernels in that layer. According to the usual practice, the parameters are set as follows:

$$\begin{aligned} k &= 1, \\ m &= 10^{-4}, \\ n &= 5, \\ z &= 0.75. \end{aligned} \quad (5)$$

- (d) Dropout should be considered a big innovation in AlexNet, which has become one of the must-have structures of neural networks. Dropout was introduced primarily to prevent overfitting. The implementation process is as follows: for those neurons of a certain layer, they are set to 0 with a probability P . Generally,

$$P = 0.5. \quad (6)$$

In this case, the dropout randomly generates the most network structure. These zero-set neurons do not participate in forward and backward propagation, as if they were frozen or discarded.

But at the same time, the number of neurons in the input and output layers is kept unchanged, and the parameters are updated according to the neural network learning method. The process is iterated repeatedly until the end of training. In this way, the entire neural network is reduced in scale and streamlined. It can also be seen as a combination of models that combine multiple models each time by generating different network structures. Thus, it can effectively reduce overfitting. Figure 5 represents a plain neural network; here, we use gray dotted circles to denote dropout neurons and use blue solid circles to indicate retained neurons.

- (e) The fully connected layer (FC) can linearly transform one feature space into another, that is, map the learned “distributed feature representation” to the sample tag space. Essentially, it implements the “classifier” function. FC tends to appear at the end to weight the previously designed features. After multiplying the input by the corresponding weight and adding an offset, they can be obtained.

The softmax layer (SL), also called the softmax function, is usually followed after the final fully connected layer. The SL works on a multiple-input multiple-output mode. The softmax function converts the value into a probability, and the node with the highest probability will be selected as the prediction target. The softmax function is defined as follows:

$$P(S_i) = \frac{\exp(V_i)}{\sum_{j=1}^N \exp(V_j)}. \quad (7)$$

3.3. Transfer Setting. In this study, only one GPU was used to implement AlexNet because of the acceptable dataset. Table 1 provides the parameters of learnable weights and biases of AlexNet. Totally, weights and biases of AlexNet are 60,965,224 (60,954,656 + 10,568). In Matlab, single-float type is chosen to store all variables. As every variable occupies four bytes, the total 233 MB space was allocated.

The neural network structure needs to be modified, especially all the last fully connected layers. The original fully connected output was designed for 1000 classifications and is not suitable for our classification topic. Thus, we redesigned the relevant layers by fine-tuning. As shown in Table 2, a fully connected layer initialized randomly containing 30 neurons was introduced, and a softmax layer and a classifier layer that can implement 30 categories were utilized. Here, the layers 23, 24, and 25 are layer indexes in the Matlab deep neural network model, which also counts the ReLU layer and pooling layer, so the indexes here are larger than indexes from a Python model.

The options of training were set by practice. Based on the principle that the training epoch of the transfer learning is relatively small, the training epoch was selected as 10. The global learning rate was set to 10^{-4} . Considering that the new layers were randomly initialized with weights and bias while the transferred layers were pretrained, the new layers learning rate were defined 10 times that of the transferred layer.

FIGURE 5: A plain neural network with dropout (gray dotted circles denote dropout neurons and blue solid circles indicate retained neurons).

TABLE 1: Learnable layers of AlexNet.

Name	Weights	Biases
CL1	$11 * 11 * 3 * 96 = 34,848$	$1 * 1 * 96 = 96$
CL2	$5 * 5 * 48 * 256 = 307,200$	$1 * 1 * 256 = 256$
CL3	$3 * 3 * 256 * 384 = 884,736$	$1 * 1 * 384 = 384$
CL4	$3 * 3 * 192 * 384 = 663,552$	$1 * 1 * 384 = 384$
CL5	$3 * 3 * 192 * 256 = 442,368$	$1 * 1 * 256 = 256$
FCL6	$4096 * 9216 = 37,748,736$	$4096 * 1 = 4,096$
FCL7	$4096 * 4096 = 16,777,216$	$4096 * 1 = 4,096$
FCL8	$1000 * 4096 = 4,096,000$	$1000 * 1 = 1,000$
Total	60,954,656	10,568

Different transfer learning settings were tried. As shown in Figure 6, setting M indicates that the layers which followed M are replaced and other layers reserved are transferred layers. Transferred layers keep learning rate as 1×10^{-4} while replaced layers initialize with learning rate of 10×10^{-4} . We will test four different transfer learning configurations: M1 to M4—M1 replaces the last FCL8; M2 replaces FCL7 and 8; M3 replaces FCL6–8; M4 replaces CL5 and FCL6–8.

3.4. Data Augmentation on Training Set. Since the deep neural network model has many parameters, the proposed model needs to contain a considerable number of sample images to achieve optimal performance. Data augmentation (DA) can extend the dataset, which helps to enhance the performance of deep learning and improve the accuracy of classification recognition. For each of the original images selected, we applied the following six DA techniques: PCA color augmentation, affine transform, noise injection, scaling, random shift, and gamma correction.

Take the “j” image as an example, which was represented in 4th column and middle row of Figure 1. As each method produced 30 new images, we gained totally 180 new augmented “j” images, which can be seen in Figure 7.

The parameters of each DA techniques were set as follows: PCA color augmentation shifted the most present color values in original images. Affine transform exerted deformation to the images and preserved straight lines. The zero-mean 0.01 variance Gaussian noise was employed to every sign language images to generate 30 new noised images. Images were scaled with scaling factor from 0.7 to 1.3

TABLE 2: Revised layers in AlexNet.

Layer	Original	Revised
23	FCL (1000) with pretrained weights and biases	FCL (30) with random initialization
24	Softmax layer	Softmax layer
25	Classification layer (1,000 classes)	Classification layer (30 classes)

FIGURE 6: Different transfer learning settings (M1 to Mn—M1 replaces the last FCL8; M2 replaces FCL7 and 8; M3 replaces FCL6–8; M4 replaces CL5 and FCL6–8.).

(a)

(b)

FIGURE 7: Continued.

(c)

(d)

(e)

(f)

FIGURE 7: Data augmentation of “j” sample image. (a) PCA color augmentation. (b) Affine transform. (c) Noise injection. (d) Scaling. (e) Random shift. (f) Gamma correction.

with step of 0.02. Then, the hand gesture image was randomly shifted 30 times with the random shift within $[-15, 15]$ pixels. Gamma correction with factor changed in the range of $[0.4, 1.6]$ with increase of 0.04.

As shown in Table 3, which denoted number of images in three partitions: training, augmented training, and test, every original image generated new training sets 181 times. The experiment was repeated ten times with data split re-setting randomly.

3.5. Training Algorithms. Deep learning often requires a lot of time and computer resources for training, so optimization algorithm is widely concerned. Adam (adaptive momentum) algorithm occupies less resources and makes the model converge faster [26], which can accelerate the learning speed and improve the effect. Adam is a first-order optimization algorithm that can replace the traditional stochastic gradient descent process. It joins the second moment estimation on the basis of momentum's first-order moment estimation and adds a moment to Adadelta. The learning rate of each parameter is dynamically adjusted by using the first and second moment estimation of the gradient. Bias correction is also added, which makes the parameters relatively stable.

The iterative formulas are as follows:

$$\begin{aligned}
 g &= (h_\theta(x^i - y^i))x^i, \\
 m_t &= \beta_1 m_{t-1} + (1 - \beta_1) * g, \\
 v_t &= \beta_2 v_{t-1} + (1 - \beta_2) * g^2, \\
 \tilde{m}_t &= \frac{m_t}{1 - \beta_1^t}, \\
 \tilde{v}_t &= \frac{v_t}{1 - \beta_2^t}, \\
 \theta_j &= \theta_{j-1} - \tilde{m}_t * \frac{\alpha}{\sqrt{\tilde{v}_t} + \epsilon},
 \end{aligned} \tag{8}$$

where g is the calculated gradient, m_t indicates the first moment of gradient g , which is also the expectation of gradient g , v_t denotes the second moment of gradient g , β_1 represents the first-order moment attenuation coefficient, β_2 represents the second moment attenuation coefficient, θ stands for the parameter that needs to be solved (or updated), and $\tilde{m}_t$ and $\tilde{v}_t$ indicate offset correction of m_t and v_t , respectively.

Two other popular training algorithms are stochastic gradient descent with momentum (SGDM) and root mean square propagation (RMSProp). In this study, we used those two algorithms as comparison basis.

SGDM, whose full name is stochastic gradient descent with momentum, introduces first-order momentum on the basis of SGD. The first-order momentum denotes the exponential moving average of the gradient direction at each moment, approximately equal to the average of the sum of the gradient vectors at the most recent T_j moment.

TABLE 3: Partition of dataset.

Partition	Number of images
Training	1,050
Augmented training	190,050
Test	270

Calculation of m_t is shown in formula above; T_j can be represented as follows:

$$T_j = \frac{1}{1 - \beta_1}. \tag{9}$$

In other words, the descending direction at time t is determined not only by the gradient direction of the current point but also by the descending direction accumulated before it. The empirical value of β_1 is 0.9, which means that the direction of decline is mainly the previously accumulated direction of decline.

On the other hand, RMSProp (root mean square propagation) is an optimization algorithm proposed by Geoffrey E. Hinton in Coursera. In order to further optimize the loss function in the update of the problem of excessive swing and further speed up the convergence of the function, RMSProp algorithm used the differential squared weighted average for the gradient of weight W and bias b . As a result, it makes greater progress in the direction where the parameter space is gentler. The sum of squares of historical gradients is smaller because of gentler direction, which leads to a smaller learning drop. Assuming that in the process of iteration t , each formula is derived as follows:

$$\begin{aligned}
 s_{dw} &= \beta s_{dw} + (1 - \beta) dW^2, \\
 s_{db} &= \beta s_{db} + (1 - \beta) db^2, \\
 W &= W - \alpha \frac{dW}{\sqrt{s_{dw}} + \epsilon}, \\
 b &= b - \alpha \frac{db}{\sqrt{s_{db}} + \epsilon},
 \end{aligned} \tag{10}$$

where s_{dw} and s_{db} are the gradient and gradient momentum accumulated by the loss function during the previous iteration $t-1$ and the vector β is an exponential of gradient accumulation. To avoid the denominator becoming 0, ϵ is going to be a very small number.

RMSProp helps to eliminate the direction of the large swing and is used to correct the swing so that the swing in each dimension is smaller. On the other hand, it also makes the network functions converge faster. RMSProp is very similar to momentum in that it eliminates the wobble in gradient descent, including minibatch gradient descent, and allows to use a higher learning rate a to speed up learning of algorithm.

4. Experiment Results and Discussion

We ran this experiment on the platform of a personal computer whose main components include 2.5GHZ Core i7

CPU, 16 GB memory, and 2 GB AMD Radeon Graphics Processor, under the operating system of Win10. Different settings of transfer learning were tried to gain optimal hyperparameters, and then we carried out 10 runs on the test set using the final model.

Average accuracy (AA) is employed to evaluate the experiment results. AA is the average of the correctly classified categories among the research objectives, which reflects the average of the accuracy of each category. Table 4 provides the average accuracy of 10 runs over the test set. In each run, 270 images were randomly selected from the entire dataset to construct the test set.

4.1. Statistical Results. As can be seen in Table 4, the results of 10 runs show that the highest average accuracy is 91.48% and the minimum value of AA is 87.78%. We marked the highest AA in bold. Additionally, it can be seen that values of AA exceed 90% in 4 times. Finally, the value of mean and standard deviation achieves $89.48 \pm 1.16\%$, which can be regarded as effective.

4.2. Training Algorithm Comparison. In this experiment, we compared Adam algorithm with SGDM and RMSProp algorithms. The comparison is shown in Figure 8.

As can be seen from Table 5, the mean and standard deviation of SGDM, RMSProp, and Adam are $88.33 \pm 2.03\%$, $89.04 \pm 0.93\%$, and $89.48 \pm 1.16\%$, respectively. We can find that Adam algorithm is superior to SGDM and RMSProp. On the one hand, the stability of Adam is more excellent than SGDM. We find that the highest average accuracy of SGDM achieves 92.22% while the lowest value of AA drops to 84.81%. There is a huge amplitude between the two, which means gradient smoothness of SGDM is not enough to achieve stable transition, that is, SGDM's inertia is not maintained enough to accommodate an unstable objective function. On the other hand, the accuracy of Adam is higher than RMSProp. Obviously, in total 10 runs, result of RMSProp has surpassed Adam only three times.

4.3. Setting of Transfer Learning Comparison. Four different transfer learning settings (configurations M1, M2, M3, and M4) were run on the test set. Table 6 and Figure 9 show the results of different settings; it can be seen that the mean and standard deviation values of configurations M1, M2, M3, and M4 are $89.48 \pm 1.16\%$, $88.93 \pm 0.75\%$, $88.59 \pm 0.90\%$, and $86.85 \pm 0.94\%$, respectively. Obviously, the greatest performance was obtained by configuration M1 (replacing the last FCL8) among all four measures. Overall, performance declines as the number of replacement layers increases. Configuration M4 (replacing CL5 and FCL6–8) achieved mean $\pm$ SD value of $86.85 \pm 0.94\%$ which is the lowest in four configurations. This situation indicates that using most transferred layers from a pretrained model is more efficacious and practical. Moreover, data augmentation expanded our dataset to a sufficient training set, which can strongly avoid overfitting and improve the accuracy.

TABLE 4: Average accuracy of 10 runs.

Run	AA
1	91.48
2	89.26
3	89.26
4	87.78
5	89.26
6	90.00
7	89.26
8	87.78
9	90.00
10	90.74
Mean + SD	89.48 ± 1.16

FIGURE 8: Comparison plot of SGDM, RMSProp, and Adam.

TABLE 5: Comparison of different training algorithms.

Run	SGDM	RMSProp	Adam (ours)
1	87.04	89.63	91.48
2	88.15	88.15	89.26
3	87.04	90.37	89.26
4	84.81	89.26	87.78
5	87.41	88.15	89.26
6	89.63	89.63	90.00
7	92.22	87.41	89.26
8	89.63	88.89	87.78
9	87.78	88.89	90.00
10	89.63	90.00	90.74
Mean $\pm$ SD	88.33 ± 2.03	89.04 ± 0.93	89.48 ± 1.16

4.4. Effect of Data Augmentation. In this experiment, we compared the performances of using data augmentation (DA) against not using DA. The experiment settings were all the same as in Section 4.1, only removing the DA on the training set. The results are shown in Table 7. We tested four different augmentation factors. First, we generate 10 new images for each method of six DAs. Therefore, we obtained

TABLE 6: Comparison of different transfer learning configurations.

Run	Configuration			
	M1 (ours)	M2	M3	M4
1	91.48	88.52	87.78	86.67
2	89.26	88.89	88.89	86.30
3	89.26	88.52	89.63	86.67
4	87.78	88.89	89.26	85.56
5	89.26	88.15	89.63	86.67
6	90.00	90.37	88.15	88.89
7	89.26	89.26	87.41	87.41
8	87.78	87.78	87.78	86.30
9	90.00	89.26	89.63	86.30
10	90.74	89.63	87.78	87.78
Mean $\pm$ SD	89.48 $\pm$ 1.16	88.93 $\pm$ 0.75	88.59 $\pm$ 0.90	86.85 $\pm$ 0.94

FIGURE 9: Error bar of average accuracies of different transfer learning configurations.

TABLE 7: Effect of data augmentation.

Run	Not using DA	Using DA 61x	Using DA 121x	Using DA 181x	Using DA 241x
1	85.19	87.78	88.15	91.48	90.37
2	85.19	85.19	89.63	89.26	89.26
3	85.93	86.30	87.78	89.26	88.89
4	82.22	86.30	85.93	87.78	89.63
5	82.59	87.41	87.41	89.26	89.63
6	84.44	86.67	88.89	90.00	89.63
7	85.56	87.04	88.89	89.26	88.52
8	83.70	85.56	90.00	87.78	88.52
9	84.07	88.15	88.52	90.00	87.78
10	83.70	87.04	88.52	90.74	90.00
Mean + SD	84.26 $\pm$ 1.24	86.74 $\pm$ 0.94	88.37 $\pm$ 1.16	89.48 $\pm$ 1.16	89.22 $\pm$ 0.79

60 new images; together with original image, we have in total 61x augmentation on the training set. Second, we set the augmentation factor as 121x. Afterwards, we also check the performance of 181x and 241x augmentation.

As can be seen from Figure 10, effect of not using DA is obviously poorer than using DA. When using DA, the performance increases with the improvement of augmentation factor. Nevertheless, 241x augmentation brings the

FIGURE 10: Comparison of DA with different augmentation factors.

FIGURE 11: Comparison plot of recent sign language classification methods.

drop of average accuracy, and it also requires the largest computation resources. Thus, setting the augmentation factor as 181x, that is, the purple bar in the diagram, gains the best performance.

4.5. Comparison to State-of-the-Art Approaches. Five other state-of-the-art approaches: HCRF [10], HMM [12], SVM-HMM [13], GLCM-MGSVM [18], and 6-layer CNN-LReLU [19], were sought to compare with proposed method in this experiment. The average accuracy of HCRF, HMM, SVM-HMM, GLCM-MGSVM, and 6-layer CNN-LReLU is 78.00%, 83.77%, 85.14%, 85.3%, and $88.10 \pm 1.48\%$, respectively, which is demonstrated in Figure 11. It can be observed that our method is superior to all the comparisons. Three attributes are helpful to enhance performance of our

method. (i) With fine-tuning, the mature pretrained model can quickly transfer previous task to our task, which is comparatively simplified. Therefore, the accuracy of our method is close to the level of the AlexNet model. (ii) The Adam algorithm can boost retraining AlexNet model, which accelerated the learning speed and improved the effect. (iii) Data augmentation extended the dataset and improved the accuracy of classification recognition. Thus, our method achieved a satisfying result.

The shortcoming of our method is AlexNet needs a massive amount of computing resources. In the future, we may try to move our algorithm to cloud computing [27,28] and mobile edge computing [29,30] areas.

5. Conclusion

In this study, an AlexNet-based transfer learning method was proposed, equipped with data augmentation and Adam optimizer. Our method was applied to fingerspelling identification for Chinese Sign Language. The experiment results demonstrated that our method achieved the average accuracy of $89.48 \pm 1.16\%$, which was excellent among the six state-of-the-art approaches. We compared three training algorithms: Adam, RMSProp, and SGDM algorithm, and found that Adam algorithm is more remarkable and stable. We tested four different transfer learning settings and discovered that configuration M1 (replacing the last FCL8) acquired greatest performance. In addition, this strategy is very practical, and it can reduce the large amount of training of the network and make full use of existing models. We also observed that using DA is more effective than not using DA, and setting different augmentation factors leads to distinct performance, in which 181x augmentation achieves the best average accuracy.

In the future, we shall try to verify different transfer learning models, such as ResNet, GoogleNet, and SqueezeNet. We need to solve the issue of setting appropriate

learning rate factor in individual layer. The dataset also needs to be further expanded for getting higher accuracy. We shall try to shift this method to other applicable areas and test other feasible methods.

Data Availability

The fingerspelling data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This study was supported by the Jiangsu Overseas Visiting Scholar Program for University Prominent Young & Middle-Aged Teachers and Presidents of China, Natural Science Foundation of Jiangsu Higher Education Institutions of China (19KJA310002), Henan Key Research and Development Project (182102310629), Natural Science Foundation of China (61602250), Jiangsu Natural Youth Science Foundation (BK20180727), Guangxi Key Laboratory of Trusted Software (kx201901), Royal Society International Exchanges Cost Share Award, UK (RP202G0230), Medical Research Council Confidence in Concept (MRC CIC) Award, UK (MC_PC_17171), Hope Foundation for Cancer Research, UK (RM60G0680), and British Heart Foundation Accelerator Award, UK.

References

- [1] X. Li, *Research on Chinese Sign Language Recognition for Middle and Small Vocabulary Based on Neural Network*, pp. 1–10, University of Science and Technology of China, Hefei, China, 2017.
- [2] M. Huber and C. Havas, “Restricted speech recognition in noise and quality of life of hearing-impaired children and adolescents with cochlear implants - need for studies addressing this topic with valid pediatric quality of life instruments,” *Frontiers in Psychology*, vol. 10, no. 2085, p. 9, 2019.
- [3] L. N. Wu, “Pattern recognition via PCNN and tsallis entropy,” *Sensors*, vol. 8, pp. 7518–7529, 2008.
- [4] L. Wu, “Classification of fruits using computer vision and a multiclass support vector machine,” *Sensors*, vol. 12, pp. 12489–12505, 2012.
- [5] G. Wei, “Color image enhancement based on HVS and PCNN,” *Science China Information Sciences*, vol. 53, pp. 1963–1976, 2010.
- [6] L. N. Wu, “Improved image filter based on SPCNN,” *Science In China Series F-Information Sciences*, vol. 51, pp. 2115–2125, 2008.
- [7] P. S. Thongpan Pariwat, “Thai finger-spelling sign language recognition using global and local features with SVM,” in *Proceedings of the 9th International Conference on Knowledge and Smart Technology (KST)*, pp. 116–120, Chonburi, Thailand, February 2017.
- [8] T.-H. S. Li, M.-C. Kao, and P.-H. Kuo, “Recognition system for home-service-related sign language using entropy-based,” *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 46, no. 1, pp. 150–162, 2016.
- [9] J. F. Lichtenauer, E. A. Hendriks, and M. J. T. Reinders, “sign language recognition by combining statistical DTW and independent classification,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, pp. 2040–2046, 2008.
- [10] H.-D. Yang and S.-W. Lee, “Robust sign language recognition with hierarchical conditional random fields,” in *Proceedings of the 20th International Conference on Pattern Recognition*, pp. 2202–2205, Istanbul, Turkey, August 2010.
- [11] G. A. Rao, P. Kishore, D. A. Kumar, and A. Sastry, “Neural network classifier for continuous sign language recognition with selfie video,” *Far East Journal of Electronics and Communications*, vol. 17, p. 49, 2017.
- [12] P. Kumar, R. Saini, P. P. Roy, and D. P. Dogra, “A position and rotation invariant framework for sign language recognition (SLR) using Kinect,” *Multimedia Tools and Applications*, vol. 77, pp. 8823–8846, 2018.
- [13] G. C. Lee, F. Yeh, and Y. Hsiao, “Kinect-based Taiwanese sign-language recognition system,” *Multimed Tools Applications*, vol. 75, pp. 261–279, 2016.
- [14] L. P. Di Wu and P.-J. Kindermans, “Deep dynamic neural networks for multimodal gesture segmentation and recognition,” *IEEE Transactions on Pattern Analysis & Machine Intelligence*, vol. 38, pp. 1583–1597, 2016.
- [15] X. Jiang, M. Lu, and S.-H. Wang, “An eight-layer convolutional neural network with stochastic pooling, batch normalization and dropout for fingerspelling recognition of Chinese sign language,” *Multimedia Tools and Applications*, vol. 78, 2019.
- [16] H. Liu, R. Cui, and C. Zhang, “Recurrent convolutional neural networks for continuous sign language recognition by staged optimization,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1610–1618, Honolulu, HI, USA, July 2017.
- [17] W. Z. J. Huang, H. Li, and W. Li, “Attention based 3D-CNNs for large-vocabulary sign language recognition,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 9, pp. 1–10, 2018.
- [18] X. Jiang, “Isolated Chinese sign language recognition using gray-level Co-occurrence Matrix and parameter-optimized Medium Gaussian support vector machine,” in *Frontiers in Intelligent Computing: Theory and Applications*, pp. 182–193, Springer, Berlin, Germany, 2020.
- [19] X. Jiang, “Chinese sign language fingerspelling recognition via six-layer convolutional neural network with leaky rectified linear units for therapy and rehabilitation,” *Journal of Medical Imaging and Health Informatics*, vol. 9, pp. 2031–2038, 2019.
- [20] M. Byra, M. Galperin, H. Ojeda-Fournier et al., “Breast mass classification in sonography with transfer learning using a deep convolutional neural network and color conversion,” *Medical Physics*, vol. 46, pp. 746–755, 2019.
- [21] C. Huang, “Multiple sclerosis identification by 14-layer convolutional neural network with batch normalization, dropout, and stochastic pooling,” *Frontiers in Neuroscience*, vol. 12, p. 818, 2018.
- [22] S. J. Pan and Q. Yang, “A survey on transfer learning,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2010.
- [23] T. Shanthi and R. S. Sabenian, “Modified Alexnet architecture for classification of diabetic retinopathy images,” *Computers & Electrical Engineering*, vol. 76, pp. 56–64, 2019.
- [24] K. Y. Son, M. E. Yildirim, J. S. Park, and J. K. Song, “Flood detection by using FCN-AlexNet,” in *Proceedings of the 11th*

- International Conference on Machine Vision*, pp. 248–255, Bellingham, Germany, November 2019.
- [25] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” *Communications of the ACM*, vol. 60, pp. 84–90, 2017.
  - [26] A. Yaguchi, T. Suzuki, W. Asano, S. Nitta, Y. Sakata, and A. Tanizawa, “Adam induces implicit weight sparsity in rectifier neural networks,” in *Proceedings of the 17th IEEE International Conference on Machine Learning and Applications*, pp. 318–325, New York, NY, USA, 2018.
  - [27] T. Wang, H. Ke, X. Zheng, K. Wang, A. K. Sangaiah, and A. Liu, “Big data cleaning based on mobile edge computing in industrial sensor-cloud,” *IEEE Transactions on Industrial Informatics*, vol. 16, pp. 1321–1329, 2019.
  - [28] T. Wang, D. Zhao, S. Cai, W. Jia, and A. Liu, “Bidirectional prediction based underwater data collection protocol for end-edge-cloud orchestrated system,” *IEEE Transactions on Industrial Informatics*, vol. 16, no. 7, pp. 4791–4799, 2020.
  - [29] Y. Wu, H. Huang, Q. Wu, A. Liu, and T. Wang, “A risk defense method based on microscopic state prediction with partial information observations in social networks,” *Journal of Parallel and Distributed Computing*, vol. 131, pp. 189–199, 2019.
  - [30] T. Wang, H. Luo, X. Zheng, and M. Xie, “Crowdsourcing mechanism for trust evaluation in cps based on intelligent mobile edge computing,” *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 10, pp. 1–19, 2019.