Journal of Computers Vol. 31 No. 6, 2020, pp. 12-23
doi:10.3966/199115992020123106002

High-Accuracy Offline Handwritten Chinese Characters Recognition Using Convolutional Neural Network

Yi Jiang[*], Yaohui Song

Department of Communications Engineering, Harbin University of Science and Technology,
Xuefu 52, Harbin, China
jasonj@hrbust.edu.cn, songyh_tx@163.com

Received 18 July 2019; Revised 11 December 2019; Accepted 20 January 2020

Abstract. Offline HCCR (Handwritten Chinese Character Recognition) is an important research area of pattern recognition. Along with the rapid development of deep learning technology, offline HCCR has been got some breakthrough. Currently, most of the HCCR solutions are either computational inefficient or not good at accuracy. A new solution is essential to provide both high accuracy and computational efficiency. In this study, a convolutional neural network based high-accuracy offline HCCR system is presented. Compared with existing HCCR solutions, the proposed solution achieves higher recognition accuracy and recall rate but uses fewer convolutional layers and fewer parameters. Experiments on the CASIA-HWDB1.1 dataset show that the recognition accuracy, top-2 accuracy, and top-5 accuracy of the proposed solution is 97.06%, 99.25%, and 99.75%, respectively.

Keywords: convolution neural network, deep learning, offline Chinese handwritten character recognition, patter recognition

1 Introduction

HCCR (Handwritten Chinese character recognition) techniques are designed to automatically recognize handwritten Chinese characters using computer. Generally, there are two approaches, online HCCR and offline HCCR. Online HCCR takes input while user is writing and recognizes the character by strokes information. Offline HCCR takes digitalized image as input. Nowadays, though offline HCCR has a wide range of applications, such as machine translation and postal automatic sorting [1], it is still the most challenging task in area of pattern recognition.

There are four reasons why offline HCCR is so difficult. The first is that there are thousands of Chinese characters. The GB2312-80 code developed in 1980 defines 6763 commonly used Chinese characters, whereas the GB18010-2000 released in 2000 defines the coding standards for 27533 Chinese characters. The number of codes defined in national standard 18010-2005 released in 2005 increased to 42711 [2]. The second is that handwritten Chinese characters are arbitrary and lack of standardization. Different people have different writing styles, so that there is lot of differences between handwritten characters by different writers. Writers may also have their own writing habits, such as continuous strokes and shorthand. Those writing habits make the handwritten characters more inconspicuous to recognize [2]. The third is that many Chinese characters have similar form. Those characters with similar form are not easy to recognize by computer. The last is that there is no dynamic information about the character in the image to be recognized. These reasons impede the accurate and efficient of offline HCCR.

In recent years, as the successful application of CNN (Convolutional Neural Network) in image classification, researchers had tried to use CNN based deep learning method to solve offline HCCR problem. For example, Zong [3] uses GoogLeNet based ensemble model to recognize handwritten Chinese characters, Yin [4] uses single CNN model to recognize handwritten Chinese characters. However, those researches don't balance accuracy and efficiency very well. Ensemble model use

[*] Corresponding Author

multiple models in parallel to get good accuracy, the huge number of parameters make it difficult to optimize and less efficient to predict. The single CNN solution uses less parameters but can only get moderate accuracy. This situation inspired us to find a more accurate and computational efficient method for HCCR.

In this study, a new CNN model named CWCCNN-V1 (Chinese handWritten Character reCognition Neural Network, Version 1) is proposed for offline HCCR. Compared with HCCR-Ensemble-GoogLeNet [3] and Multi-CNN Voting [5], which are the best two neural network models for HCCR to date, CWCCNN-V1 has higher accuracy and fewer parameters (approximately 40 million). The main contribution of this work is our model achieves higher recognition accuracy and higher recall rate with fewer convolutional layers and fewer parameters while no ensemble or voting technique was used.

1.1 Related Works

HCCR is essentially a pattern recognition task. Pattern recognition task generally involves procedures such as data preprocessing, feature extraction, dimension reduction, and classification [6]. In recent years, a lot of classic pattern recognition methods had been applied to HCCR problem. For example, [7] uses deep belief network to extract features from handwritten characters. [8] uses wavelet decomposition to extract features and uses Fuzzy Support Vector Machine as classifier. But in recent years, it looks that the application of those traditional methods meet bottleneck since no significant progress observed in offline HCCR. On the contrary, as the successful application of CNNs in image classification and object detection, researchers began to use CNN based deep learning technique to solve HCCR problems. With the help of CNN, explicit feature engineering is no longer needed, thus the original image can be simply input to the trained CNN and then recognized.

Existing neural network models for HCCR are either using ensemble model to obtain high accuracy or using simple model to get moderate accuracy. The typical ensemble HCCR models are Multi-CNN Voting [5] and HCCR-Ensemble-GoogLeNet [3]. The typical simple HCCR models are CNN-Fujitsu [4] and MQDF-THU [4].

Multi-CNN Voting [5] uses five CNNs in parallel and combine predictions from multiple models to form one final prediction. Multi-CNN Voting gets an accuracy of 96.79%. In Multi-CNN Voting, each CNN has up to 15 layers, and its total parameters far exceed any of the CNN based models in this study. Though the exact number of parameters was not given in [5], it can be estimated at 140 million according to its network structure. Although it exhibits good performance on accuracy, the huge number of parameters and voting mechanism makes it hard to train and less effective to predict. MCDNN [9] is another average ensemble model too, it composes multiple CNNs in parallel to predict.

HCCR-Ensemble-GoogLeNet [3] is another CNN model for HCCR which has a good accuracy. It ensembles four or ten base models, each of the base model is a GoogLeNet which has 22 layers and 6.8 million parameters. The HCCR-Ensemble-GoogLeNet got an average accuracy of 96.64% when using four GoogLeNet base model, and 96.74 when using ten GoogLeNet base models. Although in theoretically, as long as the base models are diverse and independent, the ensemble model always get better accuracy than its base models [10]. But the results of HCCR-Ensemble-GoogLeNet indicate that one cannot get further improvements on accuracy by ensemble more base models. So the key is to improve the accuracy of base model. Because ensemble model will always benefit from the improvements of its base model.

As stated in [11], the easiest way to improve CNN model's accuracy is increasing the depth and width of the neural network. However, this solution still has drawbacks. Firstly, as the depth and width of the neural network increasing, the over-fitting problem of the model is more likely a serious problem. Secondly, the computational resources required to train the model will rising several times. Therefore, we decided to design a neural network model for HCCR that has good accuracy yet requires fewer computing resources.

2 Convolutional Neural Networks

The offline recognition of handwritten Chinese characters is essentially a large-scale classification problem, which is a key strength of CNN. A typical CNN consists of an input layer, convolutional layer, pooled layer, fully connected layer, and output layer. The input image is subjected to convolution layer, pooling layer, activation layer, etc., and the features in the original image are extracted layer by layer, before a vector of the probability of the image belonging to a certain category is finally output by the classifier. By optimizing the feature extractor and classifier, the correct classification can eventually be achieved.

2.1 Convolutional Layer

The aim of the convolutional layer is to extract features from input. The input to the convolutional layer are several feature maps, each of which is connected to multiple convolution kernels. The input feature map is convoluted with the convolution kernel, then the feature map is obtained through the activation function. The convolution kernel is a matrix which dimensions is $k \times k$, with each value in the matrix representing a weight. The convolution operation convolves the kernel with the image pixel matrix of area covered by kernel. Each time the convolution kernel is moved on the image by a certain distance, it is moved along x-axis by m pixels and y-axis by n pixels until all candidate features on image are extracted.

Assuming that the original image I is two-dimensional, the convolution kernel is K, and the two-dimensional image convolution calculation formula is shown in equation (1):

$$S(i,j) = (I * K)(i,j) = \sum_{m}\sum_{n} I(m,n)K(i-m, j-n), \quad \textbf{(1)}$$

in which $I(m,n) \in I, S(i,j) \in S$.

After the convolution operation is performed on all feature maps, the output feature map F can be obtained by activation function, as shown in equation (2):

$$F_n^l(i,j) = f\left(\sum_{m \in M_n} \left(F_m^{l-1} * K_{mn}^l \right)(i,j) + b_n^l \right), \quad \textbf{(2)}$$

where F_n^l is the parameter of the n-th feature map of layer l, M_n is the image set connected to the n-th feature map of layer l in layer $l-1$, and K_{mn}^l is the convolution kernel that convolves the m-th feature map in layer $l-1$ and the n feature map in layer l. b_n^l is the bias of the n-th feature map of layer l. $f(\cdot)$ is the activation function.

2.2 Pooling Layer

The pooling is a down sampling strategy for CNN, so pooling layer down samples the feature map to reduce the dimension of feature map. The pooling operation uses the overall statistical characteristics around a location to represent the pixel values for that location. This method not only retains the salient features, but also reduces the feature dimensions and increases the receptive field of the convolution kernel. The maximum pooling function outputs the maximum value within the area covered by the convolution kernel. If the maximum pooling function is used, the output feature map of the pooling layer can be expressed as equation (3):

$$F_m^l(i', j') = \max\left(F_m^{l-1}(i,j) u(k,k) \right), \quad \textbf{(3)}$$

where $u(k,k)$ is the window function. Then the maximum pooling can be performed by shifting the window function over the non-overlapping subregions in feature map.

Average pooling and random pooling are commonly used in the CNN too. Pooling can lower the dimensionality of data, reduce the number of parameters, avoid over-fitting, and improve the robustness of local features to relative position changes. The neural network in this study uses max pooling to reduce the feature dimensions and improve the translation invariance of features because the handwritten image is black and white and the strokes in the image are white.

2.3 Output Layer

When using CNN for a classification problem, the logits are feed into Softmax layer for the classification output. Assuming there are K categories, and the output from the previous layer is a feature map with dimensions of $n \times 1$, then Softmax regression can be abbreviated as equation (4) [12].

$$h_\theta = \begin{bmatrix} p(y_i = 1|x_i, \theta) \\ p(y_i = 2|x_i, \theta) \\ \vdots \\ p(y_i = K|x_i, \theta) \end{bmatrix} = \frac{1}{\sum_{k=1}^{K} e^{\theta_k^T x^l}} \begin{bmatrix} e^{\theta_1^T x^i} \\ e^{\theta_2^T x^i} \\ \vdots \\ e^{\theta_K^T x^i} \end{bmatrix}, \tag{4}$$

Softmax regression can convert the input data into the probability of each category, thereby converting the classification task into a problem of finding a probability function that corresponds to each category.

3 CNN for Offline Handwritten Recognition

The proposed neural network consists of an encoder and a classifier, as shown in Fig. 1. The encoder has three convolution feature extraction modules, each of which contains two convolution layers and one maximum pooling layer. The classifier module consists of one global average pooling layer [13], two fully connected layers, and one softmax layer.

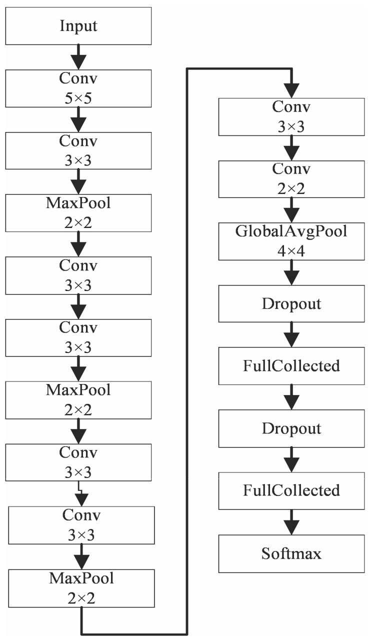

Fig. 1. Structure of CWCCNN-V1

Similar to the fully connected layer, the global average pooling also involves linear transformation to reduce the dimension of the feature map. Previous literature [13] proves that global average pooling not only replaces the fully connected layer, but also helps reduce classification errors. In addition, considering that the fully connected layer does not have a weight sharing mechanism, the number of parameters occupies a considerable proportion of the network. The global average pooling layer has no trainable parameters, so helps to reduce the number of total parameters. But in our study, global average pooling layer is used to flatten the feature maps to feature vector. In order to prevent overfitting, dropout layer is added between the global average pooling layer and fully connected layer.

The structure of the CWCCNN-V1 is shown in Fig. 1 and its network parameters are shown in Table 1. Before training the neural network, weights and biases of each layer must first be initialized. For convolutional layers in the model, weights are initialized using the random number of the Glorot normal distribution [14] and biases are initialized to zero.

Table 1. Neural network structure and parameters

Layer	Kernel Size/Stride	Output size	Number of Parameters
Conv2D	5×5/1	128×128×64	1,664
Conv2D	3×3/1	128×128×128	73,856
Max Pool	2×2/2	64×64×128	0
Conv2D	3×3/1	64×64×256	295,168
Conv2D	3×3/1	64×64×256	590,080
Max Pool	2×2/1	32×32×256	0
Conv2D	3×3/1	32×32×512	1,180,160
Conv2D	3×3/1	32×32×512	2,359,808
Max Pool	2×2/1	16×16×512	0
Conv2D	3×3/1	16×16×1024	4,719,616
Conv2D	2×2/2	8×8×1024	9,438,208
GAP	8×8/1	1x1024	0
Dropout		1×4096	0
FC		1×4096	4,198,400
Dropout		1×4096	0
FC+Softmax		1×3755	16,781,312
Total Number of Trainable Parameters			39,638,272

As shown in Fig. 1, there are four feature extraction modules in CWCCNN-V1. Each of the module contains two convolution layers and one pooling layer. The dimension of the feature map output by the last feature extraction module is $1024\times 8\times 8$.

Each convolutional layer in CWCCNN-V1 uses Leaky RELU activation function. The Leaky RELU function is one attempt to fix the dying RELU problem, it not only inherits the advantages of the RELU function, but also avoids the phenomenon that whereby neurons are suppressed when the input is negative [15]. The Leaky RELU function expression is as follows:

$$y_i = \begin{cases} x_i, & x_i \ge 0 \\ \alpha x_i, & x_i < 0 \end{cases}, \tag{5}$$

in which $0<\alpha<1$, x_i is the input of the activation function and y_i is the output.

In classification task, overfitting occurs frequently [16]. Generally speaking, overfitting is a model might be selected by maximizing its performance on some set of training data. In the proposed neural network, we introduce dropout layer between the fully connected layers to reduce the probability of overfitting in the CNN [17]. By temporarily suppressing some neurons, dropout layer reduces the co-adaptation between neurons, thereby reducing the synergy between features [18] and improving the accuracy by increasing the generalization ability. The expression for dropout is shown in equations (6) and (7):

$$r_m^l \sim Bernoulli(p), \tag{6}$$

$$a_m^l = r_m^l * F_m^l, \tag{7}$$

where r_m^l is a binary mask obeying the Bernoulli distribution, which represents the feature map processed by dropout [19]. According to the literature [17], when the dropout rate is set to 0.5, higher accuracy and faster calculation are guaranteed. Therefore, in our study, dropout rate is set to 0.5.

The dimension of the feature maps output by the 8th convolution layer is $1024\times 8\times 8$. The feature maps then converted to vector of dimension 1×1024 by global average pooling layer. This vector is the final feature vector that is ready for classification. Then, the feature vector is feed into classifier which has two fully connected layers and one dropout layer. The feature vector is converted to 4096 features by the first fully connected layer, and then 3755 features by the second fully connected layer. Finally, the 3755 features were feed into softmax layer, and then the probability corresponds to each category is generated.

4 Experiments and Analysis

4.1 Dataset and Pre-processing

This study chooses CASIA-HWDB1.1 dataset to train the neural network. The CASIA-HWDB1.1 was published by the Institute of Automation, Chinese Academy of Sciences [20]. The dataset was handwritten by 300 writers, each of whom wrote 3,755 standard Chinese characters chosen from GB2312-80. The ratio of training data to validation data is 4:1, so the training dataset has 897,758 pictures and the validation dataset has 223,991 pictures.

In order to decrease the probabilities of overfitting, data augmentation is performed on the training dataset. In our study, we have two methods of data augmentation. One is rotating image randomly, the other is adding random noise to image. After the data is augmented, there are 2,693,274 images in training dataset.

The handwritten Chinese characters in the dataset are of different resolutions. The width of handwritten images varies from 30 to 85 pixels, and heights varies from 60 to 90 pixels. So we resize all images in CASIA-HWDB1.1 dataset to 128×128. Also, we invert all images to white strokes and black ground. Four examples of pre-processed images are shown in Fig. 2. Before training, images in the training dataset are randomly shuffled, and the pixel values are normalized to prevent gradient explosion.

Fig. 2. Examples of the pre-processed images from CASIA-HWDB1.1

4.2 Experimental Setup

This study use Adam optimizer and cross entropy loss function to train the neural network model. The hardware and software used for training and validation is abbreviated as follows: Xeon E3-1285L V3, 32G DDR3 RAM, Nvidia GeForce GTX 1080Ti; Ubuntu Linux 18.04 LTS/amd64, TensorFlow-GPU 1.12. Each epoch of training takes approximate 46 minutes and the training takes 150 epochs.

4.3 Learning Rate and Loss

In this study, the learning rate is variable. As shown in Fig. 3, in the first 100 epochs, the learning rate decreases exponentially from $1e-4$ to $1e-5$. Since epoch 100, the learning rate is slightly increased to $2e-5$ for better convergence, and then decreased exponentially till the end of training.

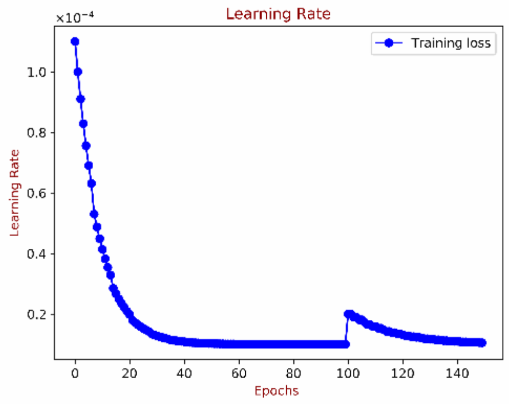

Fig. 3. Learning rate curve (training)

The loss value is shown in Fig. 4. In the first 40 epochs of training, the loss value quickly decreased from 8.0 to 1.0. As we use small batches in the training, the loss value involves relatively large noise. Therefore, we plot the moving average of the losses. The final loss of the model was 0.0675 after 150 epochs of training. As shown in Fig. 4, the proposed model converged very fast.

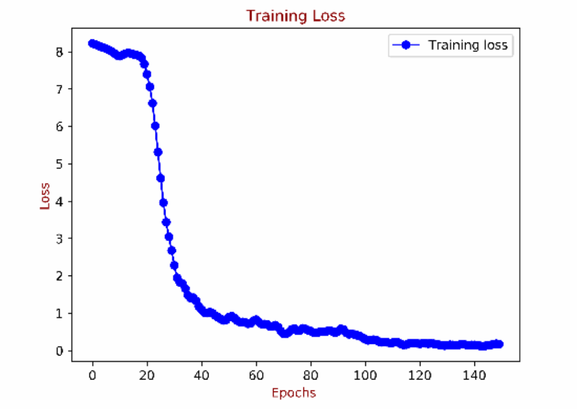

Fig. 4. Loss curve (training)

4.4 Activation Maps

The correct recognition of handwritten character images by CNNs depends on the quality of features extracted. After 150 epochs of training, the proposed model can extract features from handwritten image very well. Fig. 5 shows the original images, and the contributions of the extracted features to the recognition. The contributions are denoted by colors. Regions with higher color temperature contribute more to the recognition, regions with lower color temperature contribute less. By superimposing the contribution of all positions, we can obtain the activation map. For example, Fig. 5(A1) shows the input handwritten character image and Fig. 5(A3) shows the contribution of different positions in the input images to the recognition. Fig. 5(A2) is the result of superimposing Fig. 5(A1) and Fig. 5(A3) to illustrate where the extracted features are located in the handwritten image.

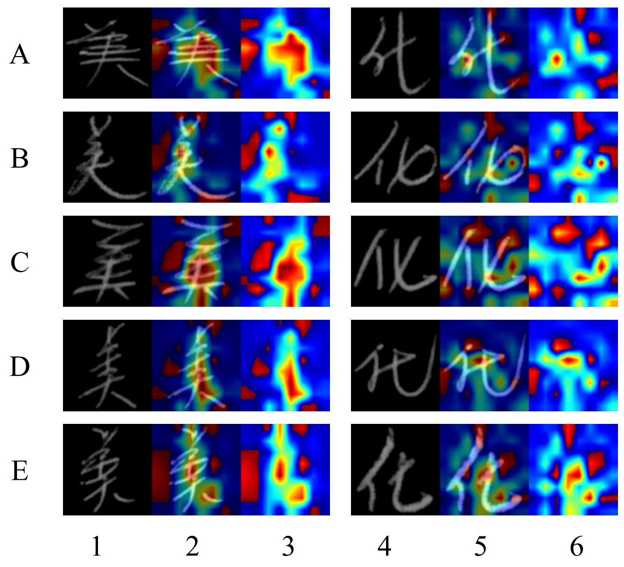

Fig. 5. Example activation map of "美" and "化"

Fig. 5 contains the handwritten Chinese characters "美" and "化" and their activation maps. Features extracted from different handwritten images of the same Chinese character coincide with the position of the font skeleton. For example, "美" is an above to below structure character, and the strokes are dense in the middle of the font. Therefore, the extracted features are more concentrated in the center, head, and tail of the glyph. "化" is a left to right structure character. The extracted features from "化" are scattered and the features at stroke endpoints and intersections are obvious. Fig. 5 demonstrates that the model can find and extract right features from handwritten character images. With the help of the right features, the proposed model can eventually recognize the characters in handwritten character images.

4.5 Accuracy

Before each batch of training, 128 and 32 handwritten images are randomly selected from the training and validation datasets respectively. The training and validation accuracy are plotted in Fig. 6. The trend of training accuracy across epochs are the same as the training loss. The training accuracy is rapidly increased from 0 to 0.6 in the first 30 epochs, then reached 0.7 after another 40 epochs of training. Finally, the training accuracy reached 0.976 at 150th epoch. Note that the training accuracy in Fig. 6 is calculated when training, so it is the accuracy on the augmented training dataset. That's why the training accuracy is slightly below the validation accuracy in Fig. 6.

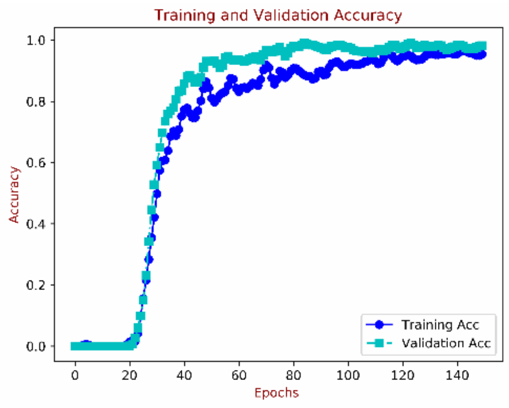

Fig. 6. Curve of Accuracy (training and validation)

According to the training loss plotted in Fig. 5 and the accuracy in Fig. 6, the proposed model can be optimized very fast. The performance of the model on validation set is shown in Fig. 7. The recognition accuracy, top-2 accuracy, and top-5 accuracy increasing steadily. After 150 epochs of training, recognition accuracy of 97.56% and top-5 accuracy of 99.75% are verified on the validation dataset. Comparing the accuracy on validation dataset (Fig. 7) and on training set (Fig. 6), we saw that the proposed model has similar performance on training dataset and on validation dataset. These results indicate that CWCCNN-V1 has better generalization performance.

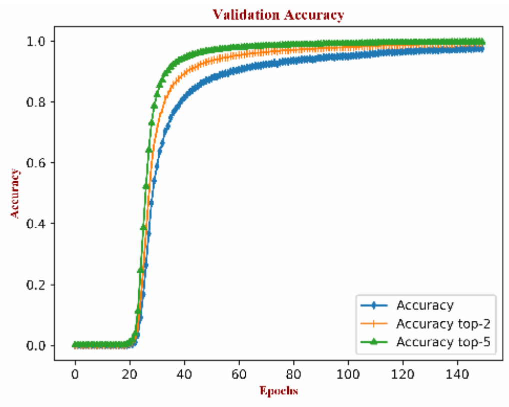

Fig. 7. Accuracy curve (on validation dataset)

4.6 Precision and Recall

The precision is a metric which describes the ability of the model to find the actual correlation in the data. Precision is calculated by dividing the number of true positives by the sum of the number of true positives and false positives. Recall is statistically understood as the ability of the model to find relevant cases in a data set and is defined as the number of true positives divided by the sum of true positives and false negatives. According to its definition, as the recall rate increased, the precision decreased, and vice versa. Therefore, the balance between precision and recall is an important metric of model performance.

In our study, we choose F_1 to present the balance of precision and recall. As defined in equation (9), F_1 is the harmonic mean of precision and recall.

$$F_1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}. \quad \textbf{(9)}$$

Precision, recall, and the other metrics based on precision and recall, such as F_1 and RoC (Receiver operating Characteristic), are defined on the binary classification model. For multi-label classification task, the classification task should be regarded as multiple binary classification task. That is, the HCCR task should be regarded as 3755 binary classification tasks so that precision and recall can be calculated. In addition, in the binary classification task, the output of the classifier is often compared with a threshold to determine the predicted output. For multi-label classification, the output always corresponds to the index with the maximum value of probability.

The precision, recall, and F_1 curves calculated at different training stages on validation dataset are shown in Fig. 8. Each point on the curve is a pair of accuracy and recall rate under different classification thresholds. The threshold is from 0.1 to 0.95 with an interval of 0.05. As the training progresses, the accuracy and recall rate curves gradually move toward the upper right corner, which reflects that the proposed model is accurate and efficient.

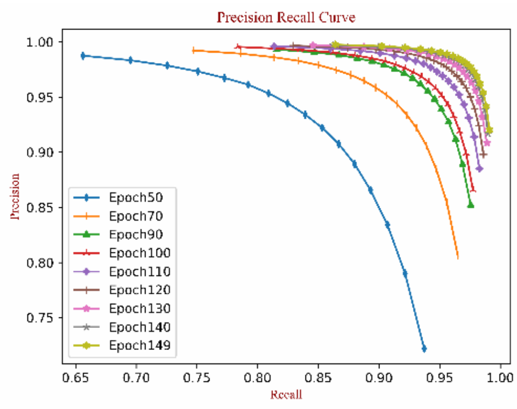

Fig. 8. Precision vs. recall (on validation dataset)

The F_1 curve at different classification thresholds and different training epochs is shown in Fig. 9. It is easy to find that the curve is almost straight after 100 epochs of training. This indicates that, after multiple epochs of training, the performance of the model is not related to the choice of threshold. Fig. 9 also indicates that the probability value of the correct classification output is large, whereas the probability value of other classification outputs is small. Therefore, the CWCCNN-V1 can recognize the handwritten Chinese character image with high confidence.

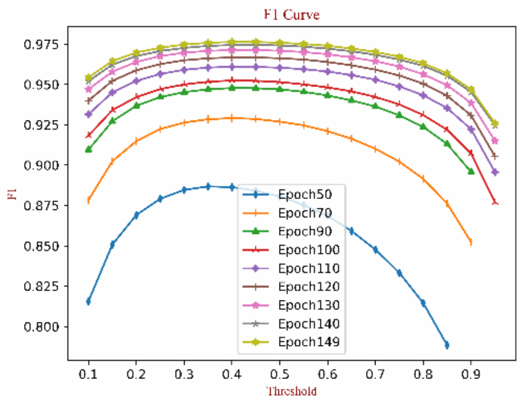

Fig. 9. F_1 curves (on validation dataset)

4.7 Receiver Operating Characteristic

The RoC curve is often used to select the best binary classification model, the RoC can also be used to select the best parameter settings that is optimal for binary classification model. Fig. 10 shows the RoC curves of the proposed model after 50, 70, 90, 110, 120, 130, 140, and 149 epochs of training. The area under the curve indicates the probability that the model correctly assigns higher scores to positive samples rather than negative samples, which reflects the performance of the classification. As shown in Fig. 10, the area under RoC curve increases gradually, indicating that the classification performance is getting better and better. Both RoC curves and F_1 curves show that CWCCNN-V1 not only recognize handwritten character images with high accuracy, but also recognize handwritten character images with high confidence.

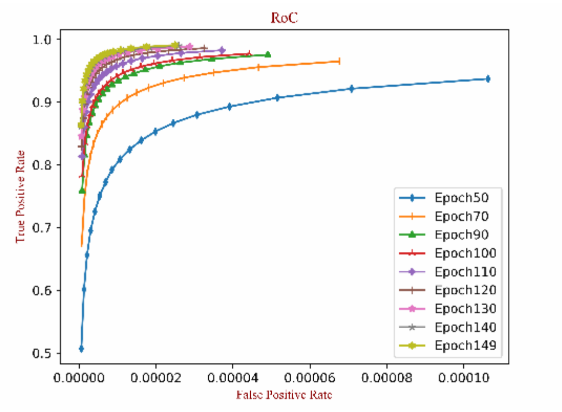

Fig. 10. RoC (on validation dataset)

4.8 Accuracy on Similar Characters Subsets

There are many similar characters in Chinese, such as ‘代’ and ‘化’, ‘句’ and ‘苟’, ‘句’ and ‘甸’. These similar characters have obvious differences in printed matter. But for handwritten Chinese characters, the writer’s individual writing style can make handwritten characters difficult to distinguish, even with the human eyes. In this study, 1553 similar characters are selected from 3,755 Chinese characters defined in GB2312-80. We use these 1553 characters to test the performance of CWCCNN-V1. These similar characters are divided into 341 subsets according to the degree of approximation of the fonts. There are average 3.55 characters in each subset. The offline handwritten Chinese character images in each subset are obtained from the validation dataset in CASIA-HWDB 1.1. The recognition is performed on each subset by CWCCNN-V1, and the results are shown in Table 2.

Table 2. Comparison of accuracy on similar character subsets

	Characters in subsets	Top1 (%)	Top2 (%)
five subsets with highest accuracy	那, 哪, 挪, 娜	100.0	100.0
	羽, 翔, 翩, 翘, 翻, 翅, 翱	100.0	100.0
	契, 楔, 揳	100.0	100.0
	旁, 榜, 膀, 傍, 磅	100.0	100.0
	辰, 唇, 晨, 震, 振, 辱	100.0	100.0
five subsets with lowest accuracy	向, 响, 晌	92.7374	100.0
	亡, 忙, 芒, 虻	91.6667	100.0
	己, 记, 纪, 妃	91.6201	100.0
	青, 清, 晴, 睛, 蜻, 猜,情,倩	90.7821	100.0
	甸, 句	90.7563	100.0

Table 2 lists the five similar character subsets with the highest accuracy and the five subsets with the lowest accuracy. One of the highest accuracy subset is “那”, “哪”, “挪”, “娜”, with an accuracy of 100%, and the lowest accuracy subset is “甸”, “句”, with an accuracy of 90.7563%. The top-2 accuracy on all subsets is 100%. The average accuracy of all 341 subsets is 95.8116%.

All the experiments results show that CWCCNN-V1 is an excellent high-accuracy offline HCCR model, it has very high accuracy on the validation dataset, also it exhibits very good performance on similar characters’ sets. The underlying reason for the good performance of CWCCNN-V1 model is that the model is good at extract features from handwritten image. As shown in Fig. 5, the model activates on the right position of handwritten character image where the strokes of character are.

4.9 Comparison with Other Recognition Models

The offline HCCR model HCCR-Ensemble-GoogLeNet [3], which is based on GoogLeNet, exhibits good accuracy of 96.64% but it is hard to training. Another deep learning model for offline HCCR is Multi-CNN Voting [5], which has an accuracy of 96.79%. It uses five CNNs in parallel and combine predictions from multiple models to form one final prediction. In Multi-CNN Voting, each base model has 15 layers, and its total parameters far exceed any of the CNN-based models in this study. Although the number of parameters is not given in [5], according to its network structure, it is approximately 140 million. In this section, models for HCCR will be compared, including the recognition accuracy and the size of model.

We compare the accuracy of various offline HCCR models on CASIA-HWDB1.1 dataset in Table 3. The accuracy of CWCCNN-V1 is 97.56% on the CASIA-HWDB1.1 dataset. According to the data in [3], HCCR-Ensemble-GoogLeNet obtains accuracy of 96.26% on the same data set. Multi-CNN Voting obtains accuracy of 96.7% [5]. CWCNN-V1 is the best among the all the HCCR models. Also, CWCCNN-V1 model surpass all the other HCCR models in top2, top5, and top10 accuracy.

Table 3. Performance comparison on CASIA-HWDB1.1 dataset

Model	Top1 (%)	Top2 (%)	Top5 (%)	Top10 (%)	Model Storage
HCCR-Ensemble-GoogLeNet [3]	96.64	n/a	99.64	99.83	110.91MB
MCDNN [9]	94.44	n/a	n/a	99.54	349MB
CNN-Fujitsu [4]	94.77	n/a	n/a	99.59	2460MB
MQDF-THU [4]	92.56	n/a	n/a	99.13	198MB
Multi-CNN Voting [5]	96.79	n/a	n/a	n/a	545MB (109MB*5)
CWCCNN-V1	97.56	99.25	99.75	99.87	151MB

The model storage is compared in Table 3. The model storage is calculated according to the neural network structure, and each parameter is a single precision floating point number (occupying 4 byte) [2]. Among all the models, Multi-CNN Voting has the largest model storage, and it is the best HCCR model except CWCCNN-V1. Compared with Multi-CNN Voting, CWCCNN-V1 has better accuracy, but only requires a quarter of storage. Compared with HCCR-Ensemble-GoogLeNet, CWCCNN-V1 needs a little more storage but has higher accuracy. For all the other models in Table 3, CWCCNN-V1 has better accuracy and requires less model storage.

5 Conclusion

In this paper, we consider the problem of offline HCCR, that is the balance between accuracy and efficiency. The existing models of offline HCCR are either good at accuracy but weak in efficiency, or good at efficiency but weak in accuracy. To address the problem, CWCCNN-V1 is proposed in this paper. The experiments show that CWCCNN-V1 exhibits excellent accuracy, top-2 accuracy, top-5 accuracy, top-5 accuracy, and recall rate. Compared with the other neural network based offline HCCR models, the CWCCNN-V1 not only has the highest recognition accuracy, but also has smaller depth and fewer parameters.

The primary limitation of CWCCNN-V1 is that it can only recognize the level-1 characters defined in GB2312-80 coding standard. But there are 13053 and 27484 characters in BIG5 and GB18030 coding standards respectively. In the future, we plan to develop a new model to recognize both traditional and simplified Chinese characters, including all the characters defined in BIG5 and GB18030 coding standard.

References

[1] Q.-X. Liu, J. Xiong, Research and application of handwritten Chinese character recognition system, Journal of Higher Correspondence Education (Natural Science) 19(4)(2006) 4-5.

[2] L.-W. Jin, Z.-Z. Yao, Y. Zhao, W.-X. Yang, Z.-C. Xie, J. Sun, Applications of deep learning for handwritten Chinese character recognition: a review, Acta Automatica Sinica 42(8)(2016) 1126-1135.

[3] Z.-Y. Zhong, L.-W. Jin, Z.-H. Xie, High performance offline handwritten Chinese character recognition using GoogLeNet and directional feature maps, in: Proc. International Conference on Document Analysis and Recognition, 2015.

[4] Y. Fei, Q.-F. Wang, X.-Y. Zhang, C.-L. Liu, ICDAR 2013 Chinese handwriting recognition competition, in: Proc. 12th International Conference on Document Analysis and Recognition, 2013.

[5] L. Chen, S. Wang, W. Fan, J. Sun, S. Naoi, Beyond human recognition: a CNN-based framework for handwritten character recognition, in: Proc. 2015 the 3rd IAPR Asian Conference on Pattern Recognition, 2015.

[6] Q.-S. Sun, J. Zhong, P.-A. Wang, D.-S. Xia, An effective combined feature extraction on handwriting Chinese character and recognition algorithm, Journal of Chinese Information Processing 19(4)(2015) 78-79.

[7] P.-P. Roy, G.-Q. Zhong, M. Cheriet, Tandem hidden Markov models using deep belief networks for offline handwriting recognition, Frontiers of Information Technology & Electronic Engineering 18(7)(2017) 978-989.

[8] C.-H. Zhu, H. Gan, J.-P. Wang, Classified identification of offline handwritten Chinese characters recognition based on FSVM, Computer Engineering and Applications 50(23)(2015) 189-193.

[9] D. Ciregan, U. Meier, J. Schmidhuber, Multi-column deep neural networks for image classification, in: Proc. IEEE Conference on Computer Vision and Pattern Recognition, 2012.

[10] V. Kotu, B. Deshpande, Data Science: Concepts and Practice, second ed., Morgan Kaufmann, 2018 (Chapter 2).

[11] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, in: Proc. IEEE Conference on Computer Vision and Pattern Recognition, 2015.

[12] R. Peng, W. Ling, X. Li, P.-W. Liu, Improved Softmax classifier for deep convolution neural networks and its application in face recognition, Journal of Shanghai University (Natural Science Edition) 24(03)(2018) 353-364.

[13] M. Lin, Q. Chen, S.-C. Yan, Network in network. <https://arxiv.org/abs/1312.4400>, 2013.

[14] X. Glorot, Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, in: Proc. International Conference on Artificial Intelligence and Statistics, 2010.

[15] Y.-S. Liu, X.-J, Wang, L. Wang, D.-J. Liu, A modified leaky ReLU scheme (MLRS) for topology optimization with multiple materials, Applied Mathematics and Computation 352(2019) 188-204.

[16] F. E. HarrellJr. Regression Modeling Strategies, Springer, New. York, 2001 (Chapter 1).

[17] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting, Journal of Machine Learning Research 15(2014) 1929-1958.

[18] G.E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, R.R. Salakhutdinov, Improving neural networks by preventing co-adaptation of feature detectors. <https://arxiv.org/abs/1207.0580>, 2012.

[19] J.-H Cheng, G.-H. Zeng, D.-K. Lu, Improved convolution neural network model averaging method based on dropout, Journal of Computer Applications 39(6)(2019) 1601-1606.

[20] C.-L. Liu, F. Yin, D.-H. Wang, Q.-F. Wang, CASIA online and offline Chinese handwriting databases, in: Proc. International Conference on Document Analysis and Recognition, 2011.