Mobile Networks and Applications (2020) 25:402–411
https://doi.org/10.1007/s11036-019-01243-5

A New Deep Learning-Based Handwritten Character Recognition System on Mobile Computing Devices

Yu Weng [1] · Chunlei Xia [1]

Published online: 29 March 2019

Abstract
Deep learning (DL) is a hot topic in current pattern recognition and machine learning. DL has unprecedented potential to solve many complex machine learning problems and is clearly attractive in the framework of mobile devices. The availability of powerful pattern recognition tools creates tremendous opportunities for next-generation smart applications. A convolutional neural network (CNN) enables data-driven learning and extraction of highly representative, hierarchical image features from appropriate training data. However, for some data sets, the CNN classification method needs adjustments in its structure and parameters. Mobile computing has certain requirements for running time and network weight of the neural network. In this paper, we first design an image processing module for a mobile device based on the characteristics of a CNN. Then, we describe how to use the mobile to collect data, process the data, and construct the data set. Finally, considering the computing environment and data characteristics of mobile devices, we propose a lightweight network structure for optical character recognition (OCR) on specific data sets. The proposed method using a CNN has been validated by comparison with the results of existing methods, used for optical character recognition.

Keywords Deep learning · CNN · Mobile computing · Optical character recognition

1 Introduction

Optical character recognition technology refers to the process of using electronic devices to scan printed characters, determine their shape by detecting edge information, and then translate the shapes into computer characters by character recognition [1]. OCR technology combines digital image processing, computer graphics and artificial intelligence and is one of the most active research topics in the field of pattern recognition. In China, there is an urgent need to use OCR technology for digital preservation of Shui characters. Shui characters are hieroglyphic, except for the Dong Ba character, which shape resembles that of an oracle bone and a gold inscription. Cultural inheritance of this language currently depends on oral communication and the handwriting of certain people; thus, most of the existing Shui characters are illegible and the books are unreadable. Therefore, by using advanced information processing methods, such as machine learning as well as big data collection and analysis, we can change the traditional document preservation methods and urgently establish meaningful digital preservation.

With the rapid development of mobile Internet services and the popularity of various intelligent devices, more and more users receive and transmit various information through mobile devices, which brings great convenience in terms of data collection, storage and use. Advances in mobile service computing and embedded devices have led to the development of the Internet of Things (IoT), which has increasingly linked physical content in everyday environments, dramatically changing the interaction between people and things. Especially in mobile phones, artificial intelligence technology represented by deep learning makes mobile phones capable of machine learning. There are many potential applications, such as object detection and recognition; speech-to-text translation; media information retrieval; and multimodal data analysis. Deep learning brings tremendous opportunities for mobile computing and

✉ Chunlei Xia
study_xia@126.com

Yu Weng
dr_wengyu@126.com

[1] College of Information Engineering, Minzu University of China, Beijing 100081, China

can greatly improve the performance of various applications. In addition, due to the rapid spread of smart portable devices and the development of mobile service technology [2, 3], the possibility of introducing smart applications in mobile environments is receiving increased attention. Therefore, people are increasingly concerned about the possibility of applying deep neural networks (DNNs) in mobile environments [4]. Deep learning not only improves the performance of existing mobile multimedia applications, but also paves the way for more complex applications for mobile devices. Many of these devices (including smart watches, smartphones, and smart cameras) can perform some sensing and processing, making them smart objects that can learn. Despite the large potential for mobile DNNs, the design of neural network architectures for mobile environments is not well developed. From basic deep learning techniques to network architecture, training and reasoning algorithms, there are still many important issues that need to be addressed.

The most widely used neural network for deep learning work is the convolutional neural network, which converts unstructured image data into structured object tag data [5]. Generally, the working principle of a CNN is as follows: first, the convolution layer scans the input image to generate a feature vector; second, the activation layer determines which feature should activate the image under inference; third, the pooling layer reduces the size of the feature vector; and finally, a fully connected layer connects each potential tag to all outputs of the pooling layer. Although current deep learning technology has made a major breakthrough in the field of OCR, the computing and storage resources of mobile intelligent devices are limited, and the convolutional neural network model usually has hundreds of megabytes of parameters, which makes it difficult to implement in mobile devices. It is a challenge to realize the unification of the interconnection and interoperability between an object and the cloud, and to guide the IoT application system that supports horizontal interconnection, heterogeneous integration, resource sharing and dynamic maintenance [6, 7]. How to abstract the capabilities provided by the "object" and "cloud" resources into a unified system of software components according to the application requirements, and define the interaction topology between these components to establish the software architecture based on IoT is also a test [8, 9].

In this article, we focus on two main research efforts. The first research work is to design a neural network structure suitable for mobile devices to enable the recognition of Shui characters. The second study aimed to design a real-time character recognition system. The proposed system uses an edge computing service paradigm and distributes data analysis throughout the network. In particular, the proposed system will split the character recognition task between the edge device (physically close to the user) and the server (typically located in a remote cloud).

2 Related works

In this section, before introducing mobile computing, let us talk about the research status of the IoT and then briefly introduce the principle of convolutional neural networks and their application in OCR; finally, let us talk about the development status of deep learning on mobile devices.

The Internet of Things is an emerging concept that is likely to change our lives as much as the Internet has. In the Internet of Things, IoT devices with smart features or objects can communicate with each other. Each device will be given a unique identifier that allows the device to access the Internet and make decisions through calculations, thus creating a new world of interconnected devices. These IoT applications are sure to make a large difference in our lives. For example, in the medical industry, HIT (Health Internet of Things) sensors can be used to monitor important patient parameters (blood pressure, heart rate, etc.), to record data to better respond to emergencies, and to provide a better quality of life for people with disabilities [10]. Another area of research based on the Internet of Things is the IoUT (Underwater Internet of Things), whose goal is to monitor a wide range of inaccessible waters to explore and protect these areas [11]. The application of interest in this article is OCR.

Optical character recognition is one of the litmus tests of pattern recognition algorithms. In addition to pattern recognition, optical character recognition involves some other fields of knowledge, such as image processing, artificial intelligence and linguistics [12–14]. Optical character recognition can make computers handle real world texts directly. The research results can be applied to a large number of practical problems, such as mail splitting, check recognition, image retrieval, intelligent robots and handwriting input. Commonly used methods in character recognition can be divided into three categories: template matching; feature extraction and classification; and deep learning methods. In the template matching method, a standard template set is first established for the character to be recognized, and then the preprocessed images of the character to be recognized are sequentially projected onto the templates for matching. The best matching template corresponds to the character being recognized. Feature extraction and recognition are the most commonly used methods in optical character recognition. The process consists of two parts. The first step uses an algorithm to extract the features of the character image. The commonly used feature extraction algorithms are: SIFT, SURF, and HOG. The second step is to use a classifier to classify the acquired features. Common classifiers include the Bayesian classifier, support vector

machine, K nearest neighbor algorithm, artificial neural network algorithm, etc. [15, 16] Deep learning has become the most commonly used method since its appearance. The most commonly used deep learning model in the field of character recognition is the convolutional neural networks.

A CNN is a feedforward neural network inspired by biological processes. Unlike traditional network architectures, a CNN contains very special convolution and down sampling layers [17]. LeNet5 was born in 1994 and is one of the earliest convolutional neural networks, which has promoted the development of deep learning [18]. This network is mainly used for handwritten character recognition. The features of the image are distributed over the entire image. Convolution with learnable parameters is an effective way to extract similar features at multiple locations with a small number of parameters. Therefore, the ability to save the parameters and the calculation process is a key development. The two main features of a CNN are local connections and weight sharing. The so-called local connection means that the nodes of the convolution layer are only connected with some nodes of the previous layer and are only used to learn local features. This kind of local connection greatly reduces the number of parameters, speeds up the learning rate, and reduces the possibility of overfitting to some extent. In addition, the convolutional neurons are grouped into feature maps that share the same weight; thus, the entire process is equivalent to convolution, and the shared weight is the filter for each map. Weight sharing greatly reduces the number of network parameters, thereby increasing efficiency and preventing overfitting. The first few layers of the convolutional network alternate between a convolutional layer and a down sampling layer, followed by a number of fully connected layers, each followed by an activation layer, resulting in a classification result. By using gradient-descent based methods and backpropagation to minimize the loss function, a CNN is trained in a similar manner to other artificial neural networks.

Since the 1990s, CNNs have gained continuous research interest. During this time, Hinton and his colleagues conducted extensive and meaningful fundamental research on deep neural networks to improve algorithmic performance and optimize architecture [19–22]. In 2012, Alex Krizhevsky proposed AlexNet, a deep convolutional neural network model, which can be regarded as a deeper and wider version of LeNet [23]. AlexNet includes several newer technical points. In the CNN, functions such as ReLU, Dropout, and LRN have been added. At the same time, GPUs have also been used to accelerate the calculations, gaining the first place in the ILSVRC 2012 competition with significant advantages. This network structure has also become a representative network structure of current convolutional neural networks. Zeiler and Fergus used random pools to tune the CNN. Simonyan and Zisserman proposed a very deep CNN to layers 16–19, achieving the greatest accuracy (ILSVRC) for ImageNet's massive visual recognition challenge [22]. The width of the network also affects the ultimate effect of deep learning. Google has proposed a deep convolutional neural network structure called "Inception" that stacks 1×1, 3×3, 5×5 convolution layers and 3×3 pooling layers. On the one hand, this increases the network's width; on the other hand, it also increases the adaptability of the network to the scale of a particular problem [24]. This network raises the level of technology for classifying and identifying the ILSVRC14 data set. He Kaiming et al. proposed a residual learning framework to reduce network training, so that each layer can learn an incremental transformation, which changes the internal network parameters and characteristics of the transmission method [25, 26]. The network is deeper than previously used networks, and training speeds have increased significantly. In March 2016, AlphaGo defeated Lee Sedol in five games. This is the first time that the Go program has defeated professional chess players [27]. It is an important milestone in artificial intelligence research.

In pattern recognition tasks, traditional machine learning tools often provide limited precision, and deep learning has been shown to be one of the most promising methods to overcome these limitations and achieve more powerful and reliable reasoning [28]. However, although deep learning technology has existed and has been successfully applied in many fields, only a few DL-based mobile applications have been produced, mainly due to the low-power computing resources provided by mobile devices [29, 30]. There are two main models for deploying DL applications on mobile devices: the client-server model and the client-only model. The former uses a mobile device as a sensor (without data preprocessing), or a smart sensor (with some data preprocessing), whose data are sent to a server or a simple cloud that runs the DL engine and sends the results back to the client. In this framework, maintaining good connectivity is the most important requirement for efficient operation of the entire system. The latter model runs DL applications directly on the mobile device. This solution can produce faster results without an operational network connection but must cope with the lack of device resources and therefore requires the right software and hardware solutions. In terms of mobile services, Honghao Gao, Wanqiu Huang and others have researched and innovated service patterns and workflows [31, 32]. In the development of deep learning, scientists have optimized model compression and neural network structure, especially for the CNN. The SqueezeNet [33] structure proposed in 2016 is not designed to achieve higher precision, but to simplify network complexity, reduce the size and calculation time of the model, and achieve performance similar to that of some typical large networks (such as Alexnet and VGG). In 2017, Google Inc. proposed a small network structure, MobileNet for embedded devices such as mobile phones, which can be applied to various tasks such as target detection, face attribute analysis and

scene recognition [34]. Similarly, there is an efficient network ShuffleNet designed for mobile devices by Vision Technology [35].

Different input data and computing environments have different structural requirements for neural networks. In this paper, our work is to design a mobile computing framework and a convolutional neural network structure based on data characteristics and mobile computing requirements to achieve Shui character recognition.

3 Method

In this section, we designed a mobile service computing framework, which is divided into three modules. And we detail the main content of Shui character recognition, which is divided into two parts: first, we describe the construction process of the data set used in the study; then, we build the CNN model and explain its details.

3.1 Mobile computing framework

Deep learning of mobile devices such as mobile phones, watches and even embedded sensors is become a hot topic. This increased interest is achieved by a growing community of academic and industrial researchers who are connecting the world of machine learning, mobile systems and hardware architecture.

On the smart mobile device, the image library is constructed according to the user's requirements, and image retrieval is realized. Most of the existing identification systems use traditional recognition technology, mainly based on image level matching, and need to manually extract features first, which makes the model susceptible to external factors such as illumination and deformation. The system extracts image features based on a convolutional neural network, which replaces the traditional artificial feature extraction algorithm and can obtain better recognition performance as it is invariant to illumination and deformation efforts. As shown in Fig. 1, the system uses a separate mechanism of training and classification. The module on the left is the data acquisition module, which collects the original data through the mobile device; the middle part is the calculation module, which performs data storage and model training in the cloud; the right part is the identification module, which completes the character recognition task. The amount of training data and the computational requirements of a convolutional neural network are extremely large, and mobile devices are over burdened by the amount of training calculations. Therefore, it is very unrealistic to expect to train the model on the intelligent device. The system first uses the GPU offline to train the lightweight image recognition CNN in the cloud; then, it loads the trained network parameters and performs the network's forward propagation operation on the mobile intelligent device, performs feature extraction, and finally searches in the image library according to the cosine similarity.

Figure 2 shows the flow of image processing by the deep learning program on the mobile device. First, an image is entered using the mobile device, either by camera or from an album. Then, the input image is detected, and the specific coordinates and size of the area to be identified are given. This step can effectively reduce external interference during recognition. Next, for the preprocessing module, the purpose is to normalize the character image to be processed to minimize the interference of angles, illumination and other factors. The key features are further located on the detected image, and the image is linearly transformed according to the position of the key point to uniformly align the feature points to the standard coordinates. Since the model input requires a grayscale image, grayscale image processing is also required. Next, the

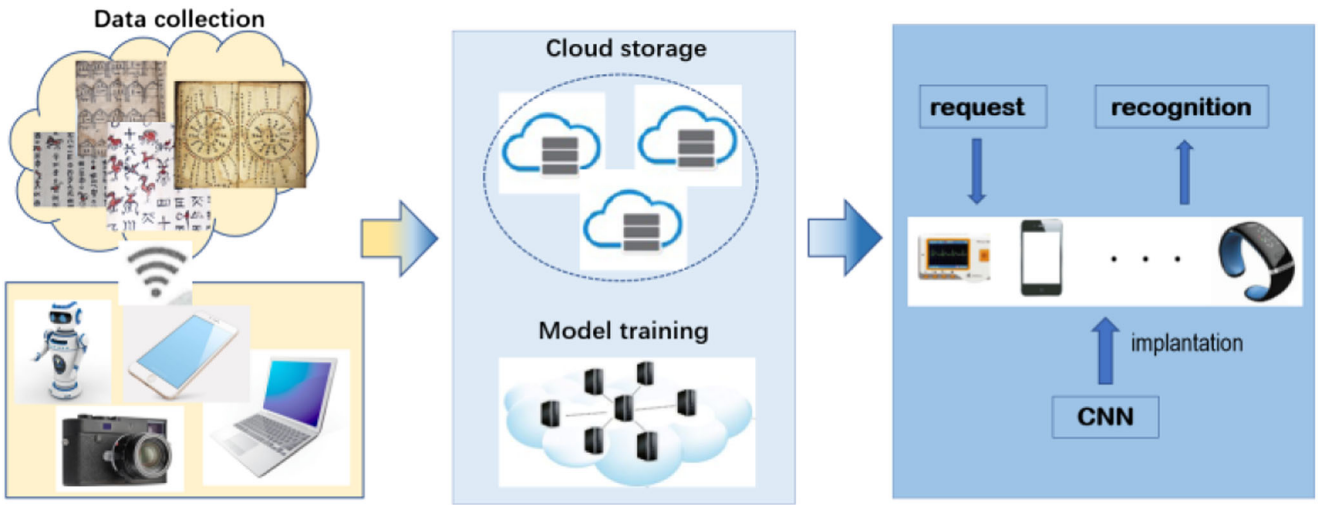

Fig. 1 Architecture of the OCR system based on IoT

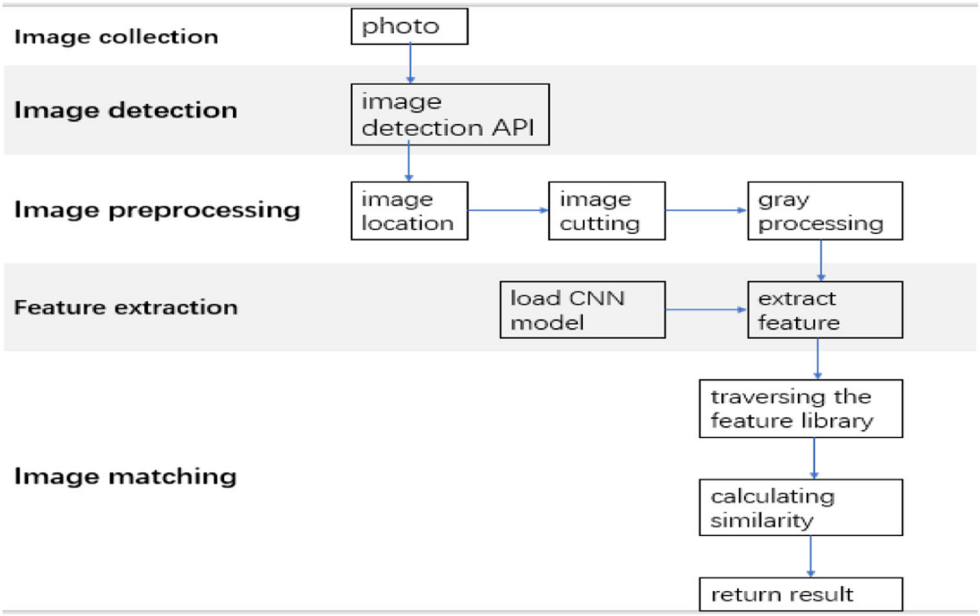

Fig. 2 Image recognition process based on mobile devices

feature extraction module first loads the CNN network parameters and then performs a forward propagation operation on the input images and extracts the output of the last layer of the hidden layer as the feature vector of the image. Finally, the retrieval module determines whether the two pictures belong to the same class by calculating the cosine similarity of the feature vectors.

3.2 Data

The raw data we collected was from a scanned file, as shown in Fig. 3.

It can be seen that the Shui character is more difficult to understand. Due to the particularity of the inheritance of the Shui character, very few people can recognize the meaning of the font. The font size is different, and there are many samples with illegible handwriting, which makes the character even more difficult to identify. We will process the collected data to create a Shui character data set.

There are several steps in the construction of the data set. First, the Shui character image is cut out from the scanned document. To prevent the loss of the character edge information during the convolution process, this research puts a blank space around each character and places the original slice in the center of the entire image. To remove noise and reduce program running time, the Shui character image size is normalized to 52*52 pixels. Third, a clustering algorithm which based on the local density of data points is implemented [36], and these image slices are used as inputs to the clustering algorithm. Finally, the class label is added to the clustering result (Fig. 4).

When marking data after clustering, professionals should be involved in the correction to avoid misclassification.

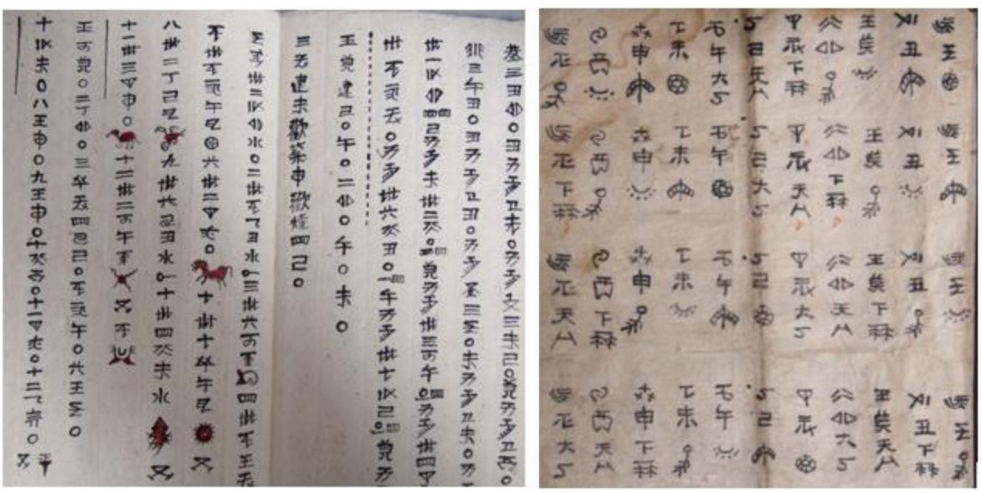

Fig. 3 Sample of Shui character

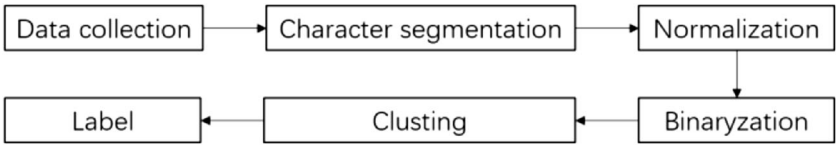

Fig. 4 Flow chart of data set construction

3.3 Structure of the CNN

With experience, the size of the input image is set to 52*52, which can be divisible by 2 as the computer's computing power permits. The image consists of many slices in the depth direction. One slice corresponds to many neurons, and the weights in the neurons can be considered convolution kernels, such as 9*9, 5*5, or 3*3 square filters. These neurons correspond to local areas in the image, and they are used to extract the features of the area. Suppose the size of the input image is W, the size of the convolution kernel is K, the movement step length of the convolution kernel is S, and P is used to fill the input image boundary; thus, the size of the convolved image is $\frac{W-K+2P}{S}+1$. Then we obtain the output tensor. The network structure we used is shown in Fig. 5.

The image size of the input layer is 52*52; in the first convolution layer, the size of the convolution kernel is 5 × 5; and after convolution, the size of the feature maps is 48 × 48. We used 128 convolution kernels; thus, the number of feature maps is 128. After the convolution layer is a max-pooling layer, and the window size of all pooling layers is 2 × 2. The feature maps we obtained after pooling were 24 × 24.In the second convolution layer, we also used 5 × 5 kernels, but the number was 192; thus, the 192 feature maps were obtained of size 20 × 20. After pooling, the size of the feature maps was 10 × 10. In the third convolution layer, we used 256 convolution kernels, whose sizes were 3 × 3, and after pooling, the size of the feature maps was 4 × 4. Then, we used 128 kernels for the last convolution layer, where the kernel size was 2 × 2; finally, we obtained feature maps sizes of 2 × 2. In the full connection layer, there were 1024 neurons, after which there was a logistic regression layer with an output of 400 because there were 400 clusters in the dataset. Table 1 shows the detailed network structure configuration.

It is well known that the choice of the activation function significantly affects the speed of convergence. Compared with traditional sigmoid function selection, the ReLU function has been proven to speed up the training process multiple times. In this study, we also noticed that convolution activation has a great influence on the description ability of the network. Driven by this observation, after trying different corrective activations, we use the ReLU function to activate each convolutional layer.

$$\mathrm{ReLU}(x)=\begin{cases} x & \text{if } x>0 \\ 0 & \text{if } x\le 0 \end{cases} \tag{1}$$

As seen from Eq. 1, the ReLU function is actually a piecewise linear function, which turns all negative values into 0, while the positive value does not change. This operation is unilaterally suppressed. Because of this unilateral inhibition, the neurons in the neural network also have sparse activating, and the negative neurons are set to 0 during the training process, which reduces the amount of calculation and is equivalent to reducing the network volume.

3.4 Training method

Here, we use the cross-entropy loss function, which characterizes the distance between the actual output (probability) and the expected output (probability). Thus, the smaller the value of the cross entropy is, the closer the two probability distributions are. Suppose the probability distribution p is the expected output, the probability distribution q is the actual output, and H(p, q) is the cross entropy; then:

$$H=-(p\ln q+(1-p)\ln(1-q)) \tag{2}$$

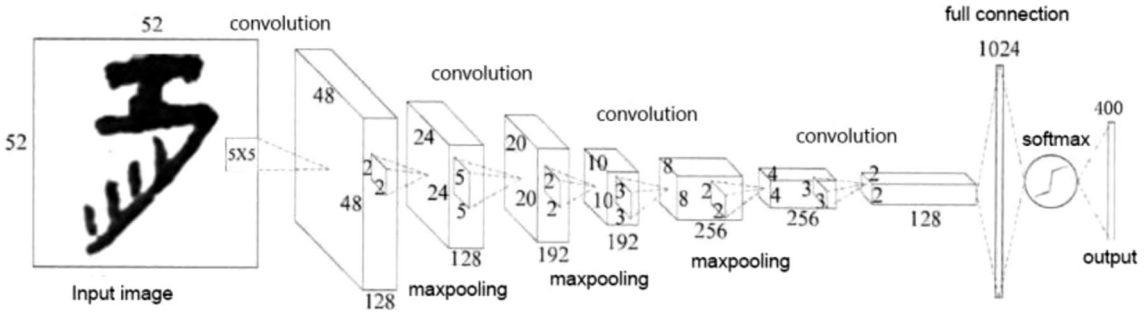

Fig. 5 Architecture of the convolutional neural network

Table 1 Overall structure of the proposed CNN

Layer	Size-out	Kernel	Parameters
Data	52*52*1	–	–
Conv1	48*48*128	5*5	3.3 k
Pool1	24*24*128	2*2	–
Conv2	20*20*192	5*5	4.9 k
Pool2	10*10*192	2*2	–
Conv3	8*8*256	3*3	2.5 k
Pool3	4*4*256	2*2	–
Conv4	2*2*128	3*3	1.2 k
Fc	1024	–	524.2 k
total			536 k

We train the network through back propagation methods. The stochastic gradient descent optimization has important practical significance in the back-propagation training process of the deep learning model. The learning rate is an important hyperparameter that controls the speed at which we adjust neural network weights based on the loss gradients.

As shown in Fig. 6, a large learning rate will cause weight updates to quickly adjust along the direction of the gradient decline, but it is easy to cause the algorithm to climb up out of a valley or cause oscillations around the optimal value such that the optimal solution is not reached. The smaller the learning rate is, the slower we progress along the loss gradient. In the long run, this cautious and slow choice may be good because we can avoid missing any local optimal solution, but it also means that we have to spend more time to converge, especially if we are at the highest point of the curve.

It is often advantageous to vary the learning rate. We use the gradient descent method that introduces the momentum to optimize the rate:

$$v = m*v - a*dw \tag{3}$$

$$w = w + v \tag{4}$$

Where v is initialized to 0, a is the learning rate, and m is a hyperparameter. Early on during the learning process, it is likely that the weights are badly wrong; thus, it is best to use a large learning rate that initially causes the weights to change quickly. Later, when approaching the optimal solution, the learning rate is adjusted to a smaller value to avoid missing the optimal value.

4 Experiment and result

4.1 Experimental preparation

We use our own Shui character dataset. First of all, we cut out the images of the Shui characters from the classic scanned documents. Second, these pictures are binarized and standardized. Third, a clustering algorithm is implemented and these image slices are used as input to the clustering algorithm. Finally, add the class label to the clustering result. After these steps, the image we get is a 52*52 binary image. Since the initial image slice is unlabeled and the convolutional neural network is supervising the neural network, we must mark this data. The data set obtained after preprocessing is shown in Fig. 7.

We experimented in the TensorFlow environment using the Python language.

4.2 Training

Among the 400 types of pictures that have been labeled, 3/4 of the 50,000 data are selected as training data and 1/4 are used as test data. To train on a convolutional neural network with 4 convolutional layers, we use the number of convolution kernels, learning rate, and batch size as variables. The results show that with the increase in the network scale, the results of convolutional neural networks are getting better, but the

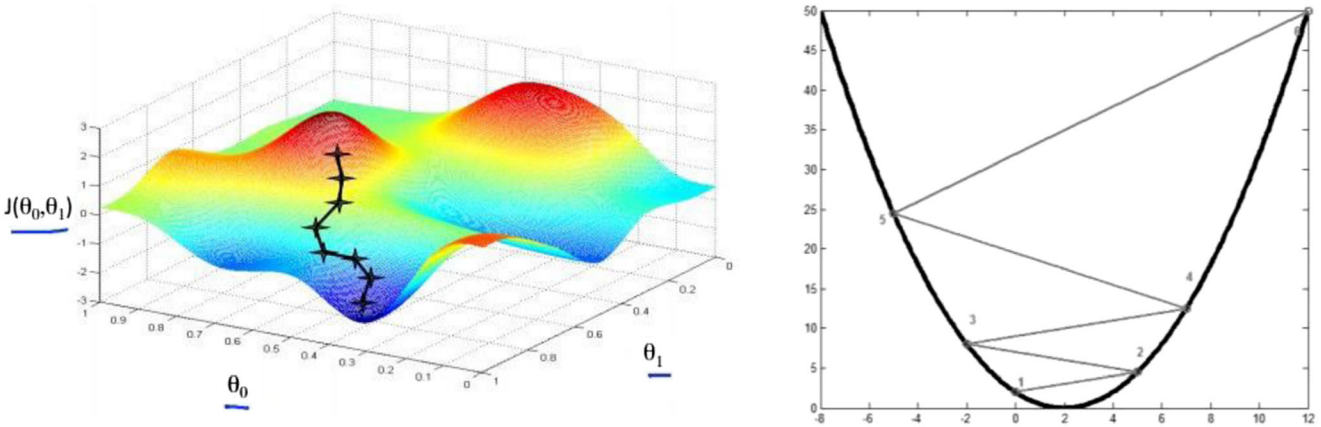

Fig. 6 On the left is the overall path of gradient descent: it shows that it is decreasing in a certain direction. On the right is shown the effect of training when the learning rate is too large

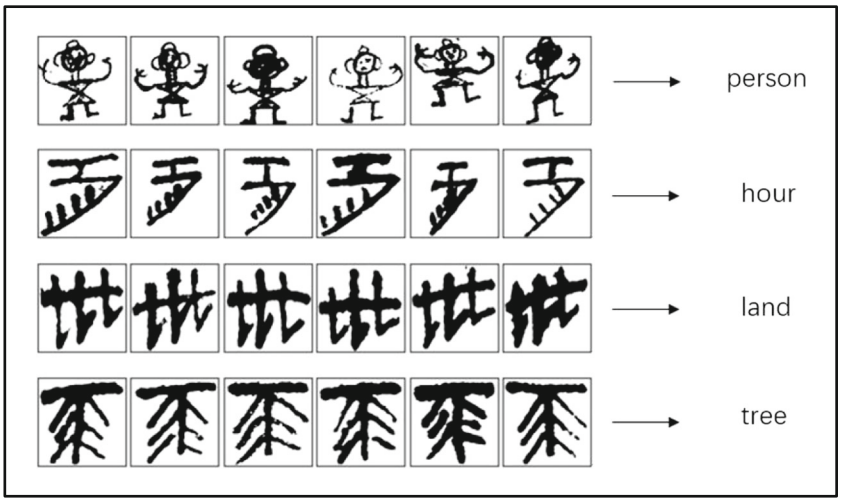

Fig. 7 Example of Shui character dataset

training time is getting longer, and there may be problems with overfitting. When the batch size is large, it is easy to find the optimal state of the model. Small batches will cause the recognition rate to fluctuate. In other words, the model only fits to a part of the data, but the batch size cannot be too large. When it increases to a certain level, the model cannot be trained, and the batch size is also limited by available memory.

When the batch changes, the learning rate has to be changed. When the batch size is too large and the learning rate is too low, the model cannot obtain sufficient training in a fixed number of iterations, while a larger batch size is insensitive to a high learning rate. In order to ensure training under the same conditions, we set the momentum value to 0.9. As shown in Fig. 8, we tested the experimental accuracy of various batch size and learning rate.

4.3 The result

In the training of convolutional neural networks, we mainly focus on three parameters: the number of convolution kernels, the learning rate, and the batch size. For the number of convolutional cores, we can slowly increase it until it reaches its optimal state. Of course, we cannot increase it all the time because it will lead to overfitting problems. For batch size, we can use relatively large numbers if resources allow, which helps us easily find the global

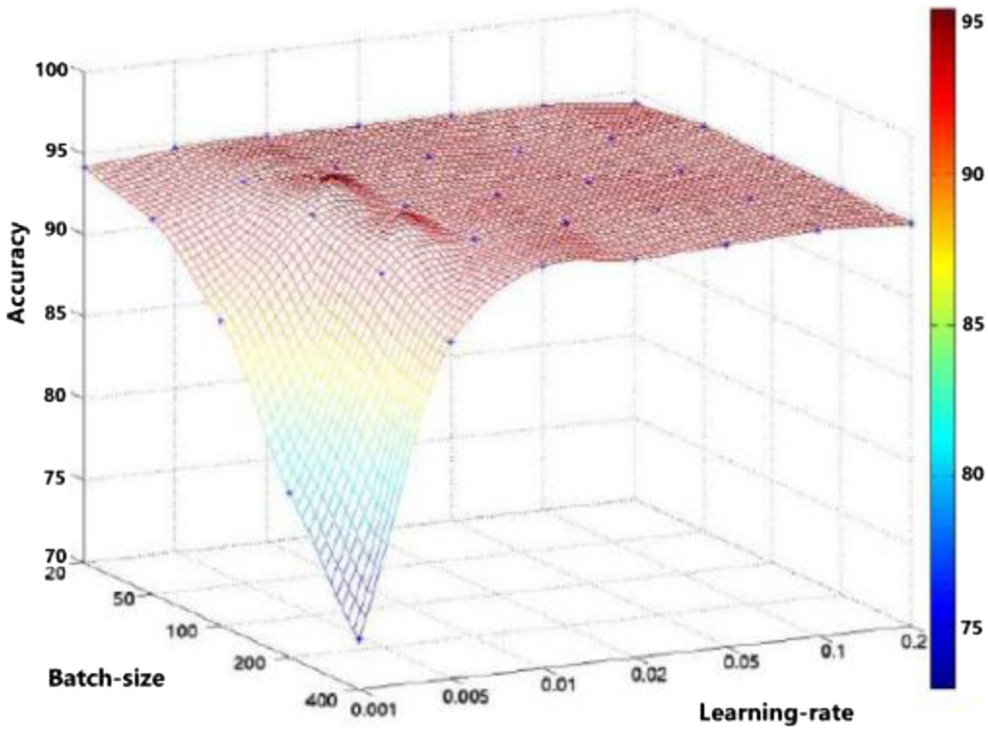

Fig. 8 The impact of learning rate and batch size on CNN

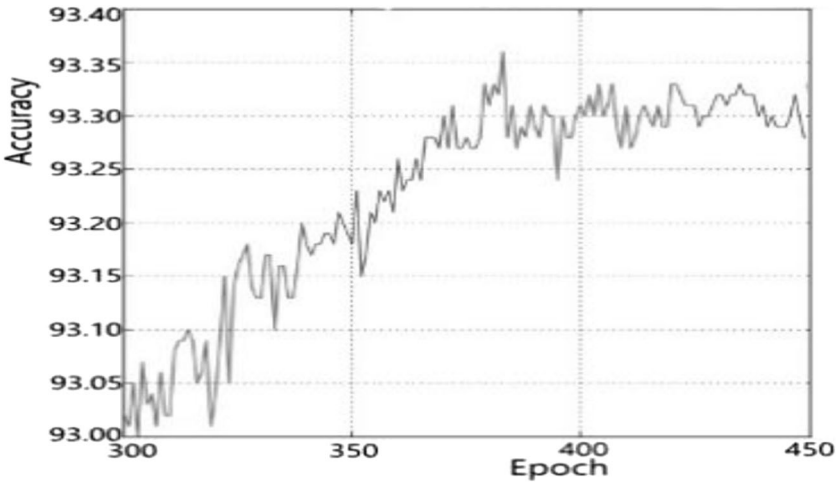

Fig. 9 The change in accuracy as the number of training increases

gradient direction. After many trials, we chose a learning rate of 0.001 and a batch size of 128.

In Fig. 9, we can see that when the training reaches a certain step size (400 epoch), the accuracy begins to oscillate, that is, the over-fitting phenomenon begins to occur, and the training should be stopped. Our final test accuracy is around 93.3%.

4.4 Comparison with others

We have also used Alexnet and Googlenet for experiments. The experimental results are shown in Table 2.

According to Table 2, Googlenet has the highest accuracy and Alexnet has the lowest accuracy. In terms of the number of parameters and the amount of calculation, our proposed network structure is significantly better than Alexnet and Googlenet. In general, the model we proposed is effective, which can achieve excellent classification performance, and is more suitable for deployment on the mobile side.

5 Conclusion

In this paper, we apply a convolutional neural network to handwritten character recognition. First, we construct a Shui character data set. Then during the training process of the convolutional neural network, we compared the results of different parameters so that we proposed the parameter tuning recommendations. The application of Shui character recognition shows that the CNN model we proposed can classify characters effectively and is more suitable for deployment on mobile devices.

Table 2 Comparison to Popular Models

model	accuracy	parameters
Our model	93.3	536 K
Alexnet	92.6	61 M
Googlenet	94.7	6.9 M

Deep convolutional neural networks have made a great breakthrough in machine learning, but there are still some research challenges. The first is the determination of the number of CNN network layers and the number of neurons, which can only rely on many experiments. Second, efficient deep learning algorithms still rely on large-scale data sets. To improve the accuracy of Shui character recognition, we also need to provide many training samples. In addition, there are many parameters in CNN, and determining the optimal parameters is also a research problem.

Furthermore, the rapid development of deep learning technology has also been widely used in the fields of virtual reality, augmented reality and mobile devices. Today, when the mobile device fully enters the "AI era", considering that CNN models are getting deeper, larger in size and more computationally intensive, it is especially important to design more efficient CNN models and compression techniques to reduce the resources (memory, storage, power, computation, bandwidth, etc.) required. At present, the deep learning model compression technology for mobile devices is mostly for a network structure. Therefore, from a software and hardware perspective, designing a complete deep learning model for mobile device deployment is still worth exploring.

Acknowledgements This work was supported by the National Natural Science Foundation of China (No. 61772575), and the National Key R&D Program of China (No. 2017YFB1402101), and the MUC 111 Project.

References

1. Mori S (1992) Historical Review of OCR Research and Development. Proc IEEE 80(7):1029–1058
2. Yin Y, Yu F, Xu Y, Yu L, Jinglong M (2017) Network Location-Aware Service Recommendation with Random Walk in Cyber-Physical Systems. Sensors 17(9):2059
3. Yin Y, Chen L, Xu Y, Wan J (2018) Location-Aware Service Recommendation With Enhanced Probabilistic Matrix Factorization. IEEE Access 6:62815–62825
4. Lane N D, Georgiev P (2015) Can Deep Learning Revolutionize Mobile Sensing? 117–122
5. Rawat W, Wang Z (2017) Deep Convolutional Neural Networks for Image Classification: A Comprehensive Review. Neural Comput 29(9):1
6. Gubbi J, Buyya R, Marusic S et al (2013) Internet of Things (IoT): A vision, architectural elements, and future directions. Futur Gener Comput Syst 29(7):1645–1660
7. Xu LD, He W, Li S (2014) Internet of Things in Industries: A Survey. IEEE Transactions on Industrial Informatics 10(4):2233–2243
8. Yin Y, Xu Y, Xu W, Gao M, Yu L, Pei Y (2017) Collaborative Service Selection via Ensemble Learning in Mixed Mobile Network Environments. Entropy 19(7):358
9. Gao H, Miao H, Liu L, Kai J, Zhao K (2018) Automated quantitative verification for service-based system design: a visualization transform tool perspective. International Journal of Software Engineering and Knowledge Engineering (IJSEKE) 28(10):1369–1397
10. Riazul Islam SM, Kwak D, Humaun Kabir M et al (2015) The Internet of Things for Health Care: A Comprehensive Survey. IEEE Access 3:678–708
11. Domingo MC (2012) An overview of the internet of underwater things. J Netw Comput Appl 35(6):1879–1890
12. Jaderberg M, Simonyan K, Vedaldi A et al (2016) Reading text in the wild with convolutional neural networks. Int J Comput Vis 116(1):1–20
13. Goodfellow I J, Bulatov Y, Ibarz J, et al (2013) Multi-digit number recognition from street view imagery using deep convolutional neural networks. arXiv preprint arXiv:1312.6082
14. Hu B, Lu Z, Li H, et al (2014) Convolutional neural network architectures for matching natural language sentences. Advances in Neural Information Processing Systems. 2042–2050
15. Vapnik VN (1999) An overview of statistical learning theory. IEEE Trans Neural Netw 10(5):988–999
16. Nasrabadi NM (2007) Pattern recognition and machine learning. Journal of Electronic Imaging 16(4):049901
17. Ciresan DC, Meier U, Masci J et al (2011) Flexible, high performance convolutional neural networks for image classification. IJCAI Proceedings-International Joint Conference on Artificial Intelligence 22(1):1237
18. LeCun Y, Bottou L, Bengio Y et al (1998) Gradient-based learning applied to document recognition. Proc IEEE 86(11):2278–2324
19. Lee H, Battle A, Raina R, et al (2007) Efficient sparse coding algorithms. Advances in Neural Information Processing Systems. 801–808
20. Salakhutdinov R, Larochelle H (2010) Efficient learning of deep Boltzmann machines. Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics:693–700
21. Zeiler MD, Fergus R (2013) Stochastic pooling for regularization of deep convolutional neural networks. arXiv preprint arXiv:1301.3557
22. Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556
23. Krizhevsky A, Sutskever I, Hinton G (2012) Imagenet classification with deep convolutional neural networks. NIPS. Curran Associates Inc
24. Szegedy C, Liu W, Jia Y, et al (2015) Going deeper with convolutions. IEEE Conference on Computer Vision and Pattern Recognition. IEEE, pp. 1–9
25. He K, Zhang X, Ren S, et al (2016) Identity Mappings in Deep Residual Networks. Computer Vision – ECCV 2016. Springer International Publishing, pp. 630–645
26. Xie S, Girshick R, Dollar P, et al (2016) Aggregated Residual Transformations for Deep Neural Networks. 5987–5995
27. Silver D, Huang A, Maddison CJ et al (2016) Mastering the game of Go with deep neural networks and tree search. Nature 529(7587):484–489
28. Lane N D, Bhattacharya S, Georgiev P et al (2015) An Early Resource Characterization of Deep Learning on Wearables, Smartphones and Internet-of-Things Devices. International Workshop on Internet of Things Towards Applications. ACM, 7–12
29. Yao S, Zhao Y, Shao H et al (2017) RDeepSense: Reliable Deep Mobile Computing Models with Uncertainty Estimations
30. Yao S, Zhao Y, Zhang A, et al (2017) DeepIoT: Compressing deep neural network structures for sensing systems with a compressor-critic framework
31. Gao H, Huang W, Yang X, Duan Y, Yin Y (2018) Towards Service Selection for Workflow Reconfiguration: An Interface-Based Computing. Future Generation Computer Systems (FGCS) 28:298–311
32. Gao H, Duan Y, Miao H, Yin Y (2017) An Approach to Data Consistency Checking for the Dynamic Replacement of Service Process. IEEE Access 5(1):11700–11711
33. Iandola FN, Han S, Moskewicz MW et al (2016) SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size
34. Howard AG, Zhu M, Chen B, et al (2017) MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications
35. Zhang X, Zhou X, Lin M, et al (2017) ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices
36. Rodriguez A, Laio A (2014) Machine learning: Clustering by fast search and find of density peaks. Science 344(6191):1492