Hindawi
Journal of Electrical and Computer Engineering
Volume 2021, Article ID 8592216, 11 pages
https://doi.org/10.1155/2021/8592216

Research Article

Research and Implementation of Fast-LPRNet Algorithm for License Plate Recognition

Zhichao Wang, Yu Jiang, Jiaxin Liu, Siyu Gong, Jian Yao, and Feng Jiang

School of Computer and Information Engineering, Central South University of Forestry and Technology, Changsha 410004, China

Correspondence should be addressed to Feng Jiang; jf09mail@126.com

Received 14 May 2021; Revised 5 July 2021; Accepted 25 October 2021; Published 20 November 2021

Academic Editor: Iouliia Skliarova

The license plate recognition is an important part of the intelligent traffic management system, and the application of deep learning to the license plate recognition system can effectively improve the speed and accuracy of recognition. Aiming at the problems of traditional license plate recognition algorithms such as the low accuracy, slow speed, and the recognition rate being easily affected by the environment, a Convolutional Neural Network- (CNN-) based license plate recognition algorithm-Fast-LPRNet is proposed. This algorithm uses the nonsegment recognition method, removes the fully connected layer, and reduces the number of parameters. The algorithm—which has strong generalization ability, scalability, and robustness—performs license plate recognition on the FPGA hardware. Increaseing the depth of network on the basis of the Fast-LPRNet structure, the dataset of Chinese City Parking Dataset (CCPD) can be recognized with an accuracy beyond 90%. The experimental results show that the license plate recognition algorithm has high recognition accuracy, strong generalization ability, and good robustness.

1. Introduction

In recent years, neural networks have been widely studied by researchers. Among them, the convolutional neural network, which has high adaptability and excellent recognition ability, has been widely used in such fields as classification and recognition and target detection [1–3]. Examples include Italian wine and iris classification [4], malicious code variant detection [5], and operational target threat assessment [6]. Because of the advantages of abundant hardware resources, powerful parallel computing capabilities, and configurable devices, FPGA has become an ideal choice for neural network implementation platforms. Therefore, many researchers have gone about studying the method of accelerating neural network by using FPGA [7–9].

There are researches on deep learning target detection algorithms at home and abroad, which not only reduces the amount of algorithm calculations and improves the efficiency of target detection in actual research applications, but also promotes the development of development platforms.

The rapid development of deep learning promotes the research process of target detection algorithms. In [10], Yi et al. introduced the adaptive strategy into the basic PNN model and proposed the adaptive neural network, which solved the problem of transformer fault diagnosis better. In [11], the learning-based intelligent optimization algorithm (LIOA) is studied, which has a certain learning ability and achieves better optimization behavior. Having studied the target detection algorithms, Safaei found that the training time of Fast-RCNN algorithm detection was reduced by 9.5 h, which opened up a new research path for target detection algorithms [12]. The advantages of the YOLO algorithm have promoted the use of YOLO in the field of license plate recognition systems [13, 14].

The rapid development of deep learning is inseparable from the support of engineering implementation technology. With the expansion of deep learning network systems, the amount of calculation of data information processing has increased, and the traditional CPU processors can no longer meet the demand for calculations. Abd E1-Maksoud et al. selected ASIC processors, FPGA processors, and GPU processors as the research objects. By comparing and

analyzing the characteristics and functions of several processors, it was concluded that FPGA processors are more suitable for development in target detection systems' application [15].

In [16], the monitoring method of neural network is used to realize the supervision and control of maglev train. In [4], an adaptive control method based on neural network is proposed to stabilize the air gap of nonlinear maglev train. All these are the combination of neural network and modern transportation system. License plate recognition is one of the important research topics in the application of target detection in the field of intelligent transportation, which has a wide range of practical application prospects [17]. Traditional license plate recognition algorithms have low accuracy and slow speed, and the recognition rate is easily affected by the environment. However, networks such as CNN and RPNet are used in license plate recognition systems with high accuracy and fast speed [18, 19]. The facts have proved that the advantages of the license plate recognition system using deep neural networks are very clear.

The research of this paper provides the following contributions:

(i) A license plate recognition algorithm, FAST-LPRNET, based on convolutional neural network was proposed

(ii) This algorithm can simultaneously complete license plate detection and segmentation-free recognition steps

(iii) The neural network hardware environment was successfully deployed on FPGA and the experiment was completed

(iv) A large number of experimental results show that this method is a fast and accurate license plate recognition algorithm

2. Overview of Convolutional Neural Networks

2.1. Network Structure. The core structure of the convolutional neural network consists of three parts, namely, the convolutional layer, the pooling layer, and the fully connected layer. The structure of the convolutional neural network is shown in Figure 1, each part of the convolutional neural network can be a single layer or multiple layers and can train images or convert data in image format. Taking the image format as input, the main function of the convolutional layer is to extract features, the function of the pooling layer is to reduce the amount of data processing while retaining useful information, and the function of the fully connected layer is to transform the feature map into the final desired output.

2.1.1. Convolutional Layer. The convolution layer convolutes the input data to extract the new feature map. After the convolution layer, the neural network does not need to deal with each point, but with each small block of regional data. Through the continuous movement of convolution kernel in the input feature map, new feature maps are obtained. These feature maps are continuously compressed, but the depth is increased so as to have a deeper understanding of the input.

In image processing, convolution operation is the weighted summation of each region by using convolution kernel. The depth represents the color image in the image to obtain the information of RGB three colors. The convolution kernel moves continuously to obtain a new feature map. Convolution kernels in convolutional neural networks are unknown and can be adjusted during training to write weight W. The calculation method of convolution layer is shown in the following formula:

$$y_j^l = \sum_{w_{ij} \in M_j} x_j^{l-1} \times w_{ij}^l + b_j^l. \tag{1}$$

Among them, y_j^l represents the output of the j neurons in the first layer of the neural network, M_j represents the set of all convolution kernels, w represents the convolution kernel, x represents the input, and b is the bias.

2.1.2. Pooling Layer. The pooling layer, also known as the downsampling layer, can achieve nonlinear dimension reduction, expand the perception field, and extract the sparse characteristics of the input. The formula of the pooling layer is shown in the following formula:

$$x_i = f\left(\beta_i \mathrm{down}\left(x_i\right) + b_i\right). \tag{2}$$

Down represents the downsampling parameters, and the commonly used pooling methods are the two ways to maximize the pooling of the local area of the pooling kernel.

2.1.3. Full-Connection Layer. The full-connection layer is also called "classifier" throughout the convolutional neural network. Each neuron in the full-connection layer is connected with the neurons in the previous layer. There is only multiplication in the full-connection layer, and one feature space is linearly transformed to another feature space to achieve the classification effect.

2.2. Main Features of CNN. CNN has the characteristics of less network parameters, abstract feature extraction, and receptive field. Because of the existence of convolution layer, the number of CNN parameters is greatly reduced. Another feature of CNN is that it has certain advantages in the extraction of abstract features. For ordinary neural networks, it is necessary to manually design certain features, and then extract features from the dataset as the input of the network. Since the extracted features are more representative, the performance of CNN is also good. The receptive field is defined as the size of the region mapped by the pixels on the feature map output by CNN at each layer on the input image, which can well reflect the correlation between the local features of the feature map.

2.3. CNN-Based Nonsegmentation License Plate Recognition Method. CNN network can be used in scene classification and image classification [20, 21]. It has strong feature

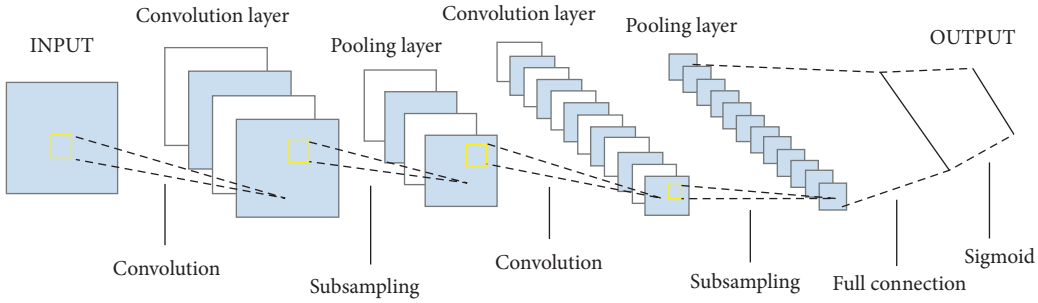

Figure 1: Structure diagram of convolutional neural network.

extraction ability, and convolution operation can maintain the spatial relationship between pixels so that the features obtained by convolution are arranged regularly. The license plate image is input into the convolution network. After a series of convolution operations, the features are arranged in a certain order. At this time, multiple full-connection layers (FC) are used to perform full-connection operation on the convolutional feature map. Through training, each full-connection layer has different sensitivity to different regions of the feature map, so as to achieve the effect of identifying multiple characters.

As shown in Figure 2, license plate Jing A12345, FC3 is responsible for predicting characters in area A3, while FC3 is basically insensitive to other areas. There are 34 different characters in region A3. To accurately identify all characters in this region, the characters appearing in the image in the training set on this region should be as complete as possible; otherwise, FC3 cannot learn all character classification features.

3. Research on the License Plate Recognition Algorithm Fast-LPRNet

Compared with the traditional license plate recognition method, undividing the end-to-end recognition method avoids the uncertainty caused by character segmentation [22]. Combined with the advantages of object-based recognition method and undivided CNN recognition method, the license plate recognition network Fast-LPRNet is designed to remove the full-connection layer and reduce the parameters, which can quickly and accurately identify the license plate.

3.1. Design of Fast-LPRNet. The Fast-LPRNet structure is shown in Figure 3. In this paper, feature extraction network is based on CNN and the classifier with weight sharing to compose the final Fast-LPRNet. Compared with the traditional license plate recognition algorithm, the end-to-end recognition network without segmentation has significant advantages in recognition accuracy and reasoning speed. The network has three convolution kernels with different structures, namely, the convolution kernel with 3×3 step length of 2, the convolution kernel with 8×1 step length of 1, and the convolution kernel with 1×8 step length of 4. ReLU is also used as a network activation function. The network

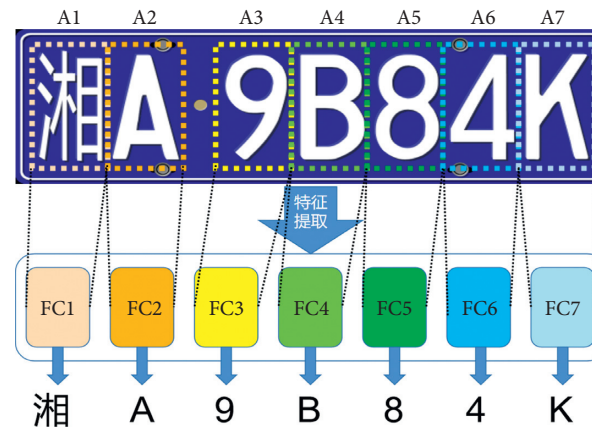

Figure 2: Nonsegmentation license plate recognition figure.

removes the pooling layer and full-connection layer, greatly reducing the network parameters and reasoning time.

The superparameters are based on experience and experiment, including learning rate and momentum. In the algorithm, the use of 3×3 step length 2 convolution can maximize the computational density and have fast computing power while having an appropriate receptive field [23]. The use of 8×1 step length 1 and 1×8 step length 4 is to change the layout of the feature graph because the feature becomes 8×32 after the 3×3 convolution. In order to facilitate and accelerate, using only one 8×1 and one 1×8 turns into a 1×7 feature map layout.

The convolution layer is used to extract the feature information of the input, which is composed of several convolution units. The parameters of each convolution unit are optimized by the backpropagation algorithm. Through the regular movement of the receptive field to the input image, the convolution operation is performed to extract the feature of the corresponding region. The bottom convolution is extracted to the low-level feature, and the high-level convolution can extract the deep feature. Convolution layer has the characteristics of local receptive field and weight sharing, which can reduce the parameters in the network. The pseudocode of the fast-LPRNet algorithm is shown in Figure 4.

The image features extracted by convolution operation are linear, but the real samples are often nonlinear. The activation function is introduced so that each pixel can be represented by any value from 0 to 1 to simulate more subtle

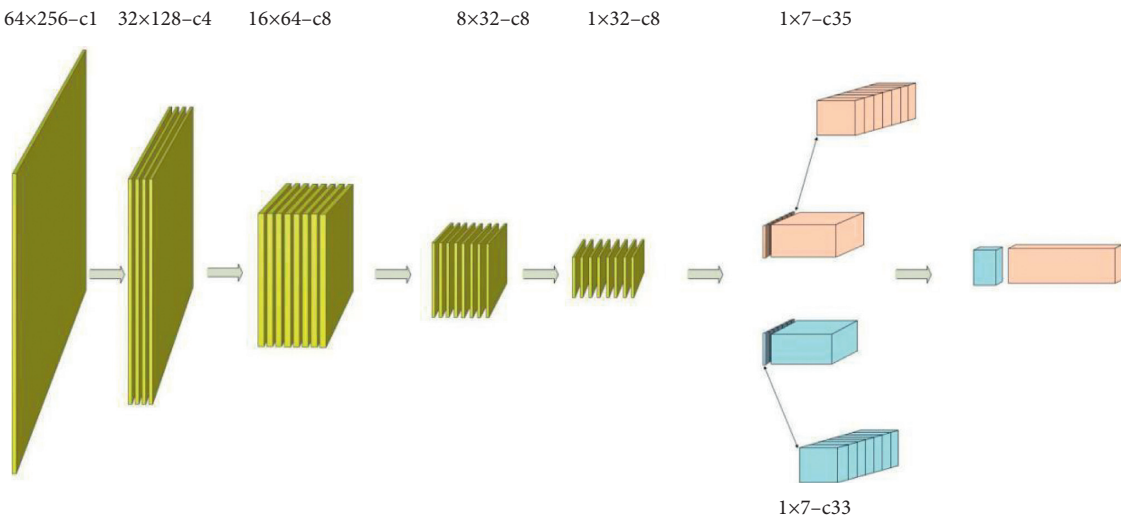

Figure 3: Fast-LPRNet structure diagram.

Fast-LPRNet algorithm

Input	*h*×*w* image size, *channel* image channel, *weight* paramter, R*elu* function
1	Initialize i=0, j=0, height × width convolution kernel, *Feature, Feature* temporary variable *temp*
2	Normalized *h*=64 picture width, *w*=256 picture height
3	Repeat for *i* to *height* for *j* to *width* *temp*=∑ conv (*weight, h, w*) *Relu* Read the temp Final *w*=1, *h*=7, *channel1*=33, *channel2*=35
4	Select the character image classifier according to the channel1 and channel2 parameters
5	Load the *weight*, and all the feature maps are accumulated and added to get the *Feature* value
6	Obtain the maximum value index according to the *Feature*, judge whether it is the same, if it is, the correct recognition result
7	Calculate the probability of recognition
Out	Print recognition results and recognition rate

Figure 4: Pseudocode of fast-LPRNet algorithm.

changes. Activation functions are generally nonlinear, continuously differentiable, and monotonic. The network uses the ReLU function as the activation function. Compared with the sigmoid function and the tanh function, the ReLu function has the characteristics of unilateral suppression and sparse activation [24]. That is, when $x > 0$, the gradient is constant to 1, and there is no gradient dissipation problem. When $x < 0$, the output is 0. After training, the more neurons 0, the sparser the network, and the more representative the extracted features can alleviate the problem of overfitting to some extent. The disadvantage of this activation function is that the forced sparse processing will cause the model to fail to learn more effective features, so pay attention to the setting of the learning rate during training to prevent too many neurons' necrosis.

3.2. Implementation of Fast-LPRNet

3.2.1. Shared-Weight Classifier. Two shared-weight classifiers with convolution kernel of 1×8 step size of 4 are used to perform convolution operation on the feature map, and two feature maps with size of 1×7 are obtained. Their channel numbers are 33 and 35, respectively. The layout of the 1×7

feature map is just in line with the layout of seven characters in the license plate. The 33 channels correspond to 32 Chinese characters and one unidentified character of the license plate. The 35 channels correspond to 24 letters plus 10 numbers and one unidentified character on the license plate. Classifier1 is responsible for recognizing the Chinese characters of the license plate; classifier2 is responsible for recognizing the remaining characters, and then combining these two results to get the final recognition result.

The fully connected layer classifier is a commonly used classifier. It summarizes the results of the convolutional layer, pooling layer, activation function, etc., and performs work similar to template matching again, abstracting the probability of the existence of the feature of the number of fc neurons size; then it goes through a layer of fc and then classifies the features output by the previous layer of fc to obtain each feature. Generally speaking, the original license plate image is extracted by convolution layer and then classified by full-connection layer. However, the full-connection layer classifier also has defects, such as the balance of the dataset affects the performance of the classifier, and the operation speed is slow due to the large number of parameters. This can be avoided by using the right reshared convolution layer instead of the classifier, because each convolution classifier is a complete convolution process on the license plate and has the same interest in the seven regions on the license plate. At the same time, because of the weight sharing, the number of parameters is less than the full-connection layer, and after a convolution, it can output six-character prediction results, and the reasoning speed is much faster than the full-connection layer.

3.2.2. Convolution Layer instead of Pooling Layer. The convolution kernel with step length of 2 is used to achieve the purpose of downsampling, replacing the role of the pooling layer. Based on ResNet, the convolution layer with step size of 2 is used to replace the pooling layer with size of 2 to realize the downsampling operation [25]. The main significance of pooling layer is invariant, which includes translation invariance, scale invariance and rotation invariance. At the same time, pooling downsampling makes the high-level features have larger receptive fields. Experiments show that invariance has a small effect on performance, while expanding receptive field can be replaced by CNN + ReLU with stride = 2, and the performance can be basically consistent or even slightly better. For the pooling layer and the convolution layer with step size of 2, the pooling layer is a prior downsampling method; that is, it is believed that the downsampling rules are determined. For the convolution layer of stride = 2, its parameters are obtained by learning, the sampling rules are uncertain, and it is possible to learn more important features. The convolution diagram is shown in Figure 5.

3.2.3. Training of Fast-LPRNet. The network was trained in Python 3.8 environment, with CUDA version 10.1 and PyTorch version 1.7. The parameters saved after the training were parsed and turned into a format that can be read by programs executed on Intel Cyclone V SoC. The pth file saved by PyTorch stores parameters in the form of key-value pairs, uses python to process the key-value pairs, extracts the weights of each convolution kernel separately, and stores them in a txt file in a certain order as floating-point numbers. The program executed on Intel Cyclone V SoC gets the weight of each convolution kernel by reading the txt file line by line and stores the weight in the corresponding array for convolution operation call.

4. Experiment and Discussion

In order to verify the feasibility of the improved algorithm, the proposed algorithm is applied to the hardware experimental platform built by FPGA. The platform uses hardware and software cooperation to verify the license plate recognition. Firstly, HLS high-level synthesis tool is used to complete the design of CNN algorithm acceleration [26], and then Quartus integrated development environment is used to complete the hardware design. The Linux system is built in PS of FPGA, and the dataset, executable program, and device tree file are copied into SD card to start Intel FPGA: Cyclone V development board for license plate recognition. The specific experimental platform architecture is shown in Figure 6.

4.1. Implementation of License Plate Recognition Based on FPGA. The operating environment of this experiment is CentOS Linux system, in which Quartus Prime 18.1 is the FPGA development software, Eclipse is the integrated development environment, C language is used for development, and Cyclone V: 5CSEBA6U23I7 is the selected FPGA chip model.

In Linux system, HLS compiler tool $i++$ is used to compile the conv layer and ReLu layer code written by *c* into RTL level, which is added to the Q-SYS system. The required clock and reset interface are configured, and the four operators are connected to the control IP through the Avalon interface and HLS interface.

According to the structure of the neural network, three convolution kernels with different structures are needed, namely, the convolution kernel with 3×3 step size of 2, the convolution kernel with 8×1 step size of 1, and the convolution kernel with 1×8 step size of 4. In addition, ReLU is used as the activation function of the network, which is also the key part that needs to be accelerated. The IP of the four operators compiled by HLS is shown in Figure 7. The hardware structure of the corresponding neural network structure is designed in Quartus Prime, and the generated sof file is generated to generate the .rbf file required by uboot, and finally the .dtb binary device tree file required by Linux is generated.

4.2. Test and Analysis

4.2.1. Graying. Converting a color image to a grayscale image refers to image decolorization. Grayscale operation can simplify the image information, but the recognition

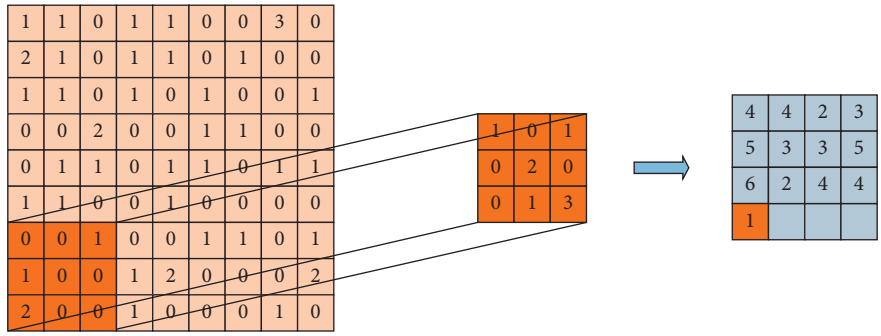

Figure 5: Convolution diagram (3 × 3) (step 2).

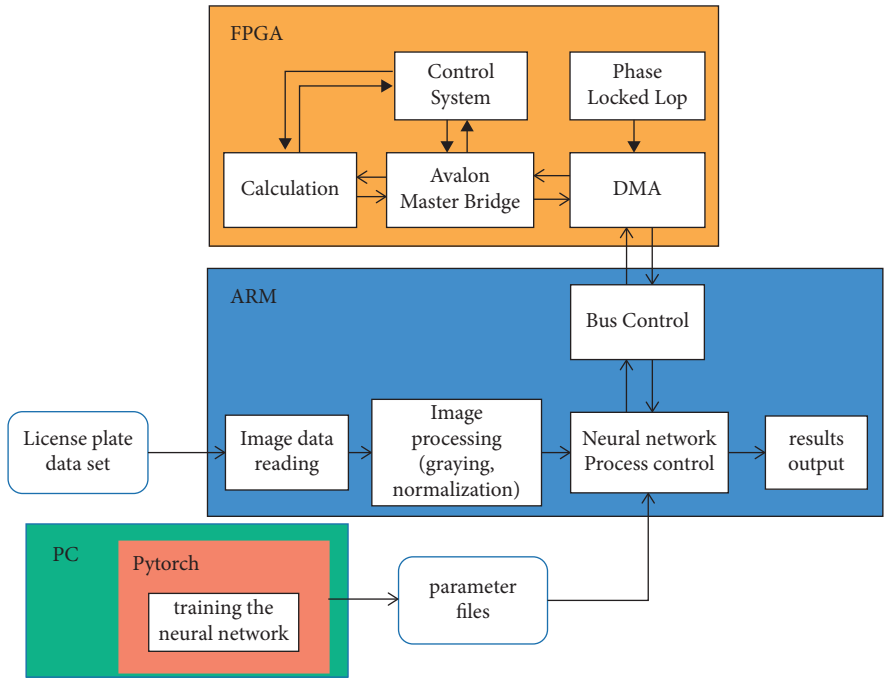

Figure 6: System framework diagram.

accuracy is not affected. Color image has three components. If red, green, and blue are set as the same value, the image will become gray image, that is, monochromatic image. The gray value is represented by the number between 0 and 255, representing the gray depth between white and black. The results of license plate graying are shown in Figure 8.

4.2.2. Normalization. Before the image is input to the convolutional neural network, the convolutional neural network is standardized to make different sizes of data fall in a small specific interval. The most typical normalization method is used to map the data to [0, 1] interval. Normalized image data has the following advantages:

(1) Improving the convergence speed of the model: when the value range of two different feature objects is too large, when optimizing them, an irregular narrow and long ellipse will be obtained. When training on the network, the direction of gradient descent will change from the direction of the vertical contour is changed to a zigzag route, which greatly reduces the iteration speed and affects the efficiency of network recognition.

(2) Improving the accuracy of the model: when the network needs to perform distance calculations such as Euclidean distance, the difference of the features will cause the loss of accuracy.

4.2.3. System Performance Test. The experiment builds a test environment on the AWCloud artificial intelligence edge experimental platform to test the detection function and operation performance of the license plate recognition algorithm. The LPR network uses an "end-to-end" recognition method. For the dataset of 30 license plate images, Fast-LPRNet was used for recognition. Thirty license plate images were successfully identified within 5.5 seconds, and the recognition accuracy was 100%. The recognition results are shown in Figure 9.

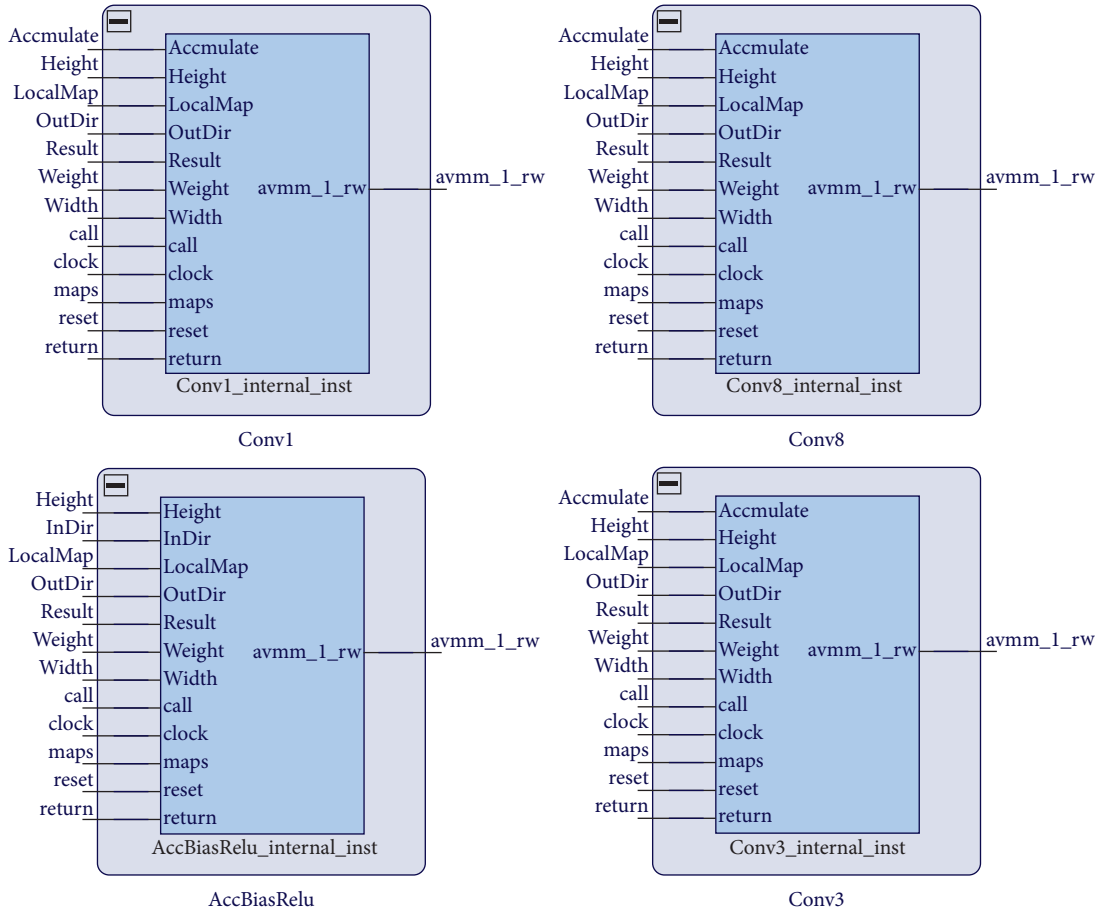

Figure 7: Convolution kernel IP and ReLu function IP core.

Figure 8: Graying of license plate.

4.2.4. Algorithm Generalization Ability Test. The recognition method in this paper eliminates the segmentation step enough to extract feature information on the complete license plate image so that the network has strong generalization ability. In order to test the generalization ability of the network, add some interference to the license plate image, test the accuracy of the network to identify the license plate. As shown in Figure 10, adding random salt and pepper noise in the case of character adhesion, the network can correctly identify.

In addition, different interference is added between the upper, lower, and diagonal characters of the license plate image. As shown in Figure 11, after testing, the network can still accurately identify the license plate characters.

4.2.5. Algorithm Scalability and Robustness Test. In order to explore the high scalability of the proposed license plate recognition network, the depth of the network is increased on the basis of the Fast-LPRNet structure, and the Deep-LPRNet license plate recognition network is obtained. The structure is shown in Figure 12. Deep-LPRNet has more convolution layers and channels and can extract more feature information from the original image. There is no additional block structure in the network, such as BN layer and residual block, because it is necessary to analyze the expansion ability of the end-to-end identification network itself, so it is necessary to have a network structure that is basically the same as Fast-LPRNet, additional block structure that reduces the network overfitting ability and convergence effect.

Deep-LPRNet is used to train CCPD data. CCPD dataset is a public dataset of Chinese automobile license plate, which is commonly used in the research of license plate recognition [27, 28]. At present, there are more than 250,000 license plate images, including a total of nine categories, as shown in Figures 13 and 14 shows the sample images in the CCPD dataset.

Select CCPD-Base images after cutting and resize (Figure 15) to network training. CCPD-Base has nearly 200,000 images, selecting 120,000 as training sets and 80,000 as test sets, using the PyTorch framework for training. Other hyperparameters are basically the same as when training Fast-LPRNet. When the network is trained to the convergence state, the recognition accuracy reaches more than 90%

```
--------------------
鄂REM1S7.BMP read and resize the success
Identify the results :  鄂REM1S7.BMP
--------------------
新PZ5603.BMP read and resize the success
Identify the results :  新PZ5603.BMP
--------------------
黑VE9K0Y.BMP read and resize the success
Identify the results :  黑VE9K0Y.BMP
--------------------
Successfully identified 30 . Total of 30 .
Accuracy = 100.00%
run time 5.344946 s

real    0m5.381s
user    0m5.300s
sys     0m0.050s
root@awcloud:/home#
```

Figure 9: System identification result.

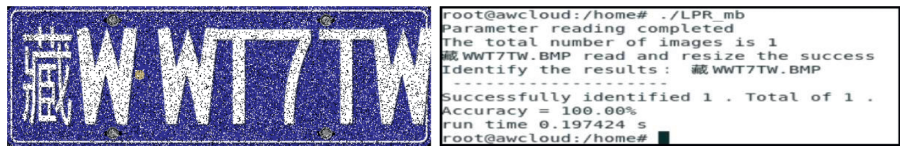

Figure 10: Salt and pepper noise interference and recognition results.

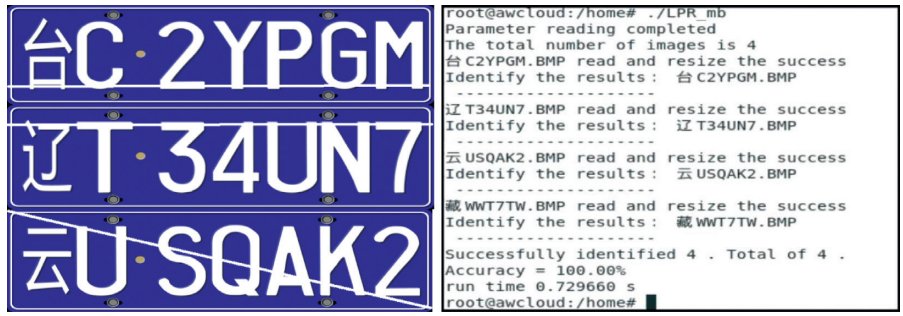

Figure 11: Interference and recognition results of three different parts.

Deep-LPRNet

size	channel	description
64×256	3	CONV3-ReLu
32×128	16	CONV3-ReLu-CONV3-ReLu
16×64	64	CONV3-ReLu-CONV3-ReLu
8×32	128	CONV3-ReLu-CONV8×1-ReLu
1×32	128	{CONV1×8}×2
1×7	33	out
1×7	35	out

Figure 12: Deep-LPRNet structure.

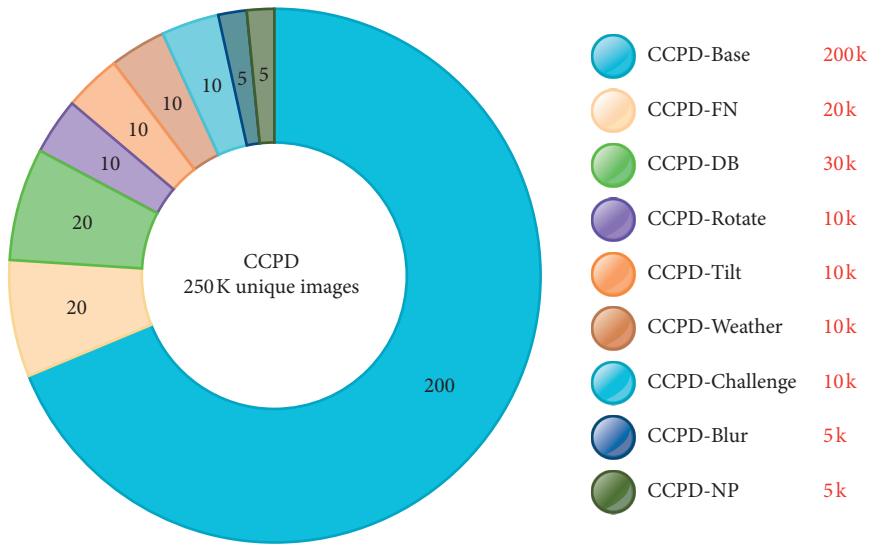

Figure 13: CCPD dataset layout [14].

Figure 14: Sample picture of CCPD dataset [14].

Figure 15: License plate images after cutting and adjusting size.

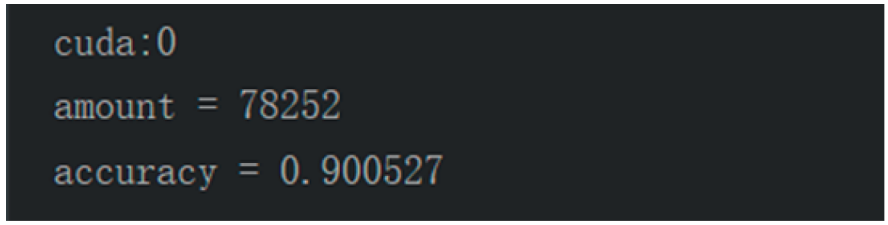

Figure 16: CCPD recognition results.

on the test set (Figure 16), indicating that the network based on this structure has strong scalability and robustness and has strong practical application value.

5. Conclusion

This paper studies the algorithm of license plate recognition network based on FPGA. Through the combination of software and hardware design, the Fast-LPRNet network is proposed on the basis of CNN, and the license plate segmentation is improved and optimized. Finally, the feasibility of the algorithm is verified. The average frame rate is 5.45 frames/second, and the accuracy of recognizing 30 license plate images is 100%. The deep-LPRNet network is added. The recognition test is carried out on the CCPD-Base dataset, and the recognition accuracy is as high as 90%.

In this study, the combination of convolutional neural network and FPGA development is our innovation, but there is still room for improvement in FPGA hardware. In the future work, we hope to optimize the algorithm structure and build the optimized hardware structure on the FPGA. In addition, it will be combined with some other new meta-heuristic algorithms, such as monarch butterfly optimization (MBO) [29], earthworm optimization algorithm (EWA) [30], elephant herding optimization (EHO) [31], moth search (MS) algorithm [32], Slime mould algorithm (SMA) [33], Harris's hawks optimization (HHO) [34], and so on. This combination will further improve the performance of Fast-LPRNet.

Data Availability

The algorithm recognition accuracy data used to support the results of this research are included in this article.

Conflicts of Interest

The authors declare that they have no conflicts of interest regarding the publication of this study.

Acknowledgments

This study was funded by Changsha Science and Technology Project (no. kq1901139) and Science and Technology Project of Hunan Province (no. 2016WK2023).

References

[1] M. W. Lung, C. W. Chun, C. T. Wang, and Y. H. Lin, "Recycling waste classification using optimized convolutional neural network," *Resources, Conservation and Recycling*, vol. 164, Article ID 105132, 2021.

[2] S. T. Sara, M. M. Hasan, A. Ahmad, and S. Shatabda, "Convolutional neural networks with image representation of amino acid sequences for protein function prediction," *Computational Biology and Chemistry*, vol. 92, Article ID 107494, 2021.

[3] N. Nahla, E. Mohammed, and V. Serestina, "Face expression recognition using convolution neural network (CNN) models," *International Journal of Grid Computing & Applications*, vol. 11, no. 4, pp. 1–11, 2020.

[4] Y. Sun, J. Xu, G. Lin, and N. Sun, "Adaptive neural network control for maglev vehicle systems with time-varying mass and external disturbance," *Neural Computing and Applications*, pp. 1–12, 2021.

[5] Z. Cui, F. Xue, and X. Cai, "Detection of malicious code variants based on deep learning," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 7, pp. 3187–3196, 2018.

[6] L. Wang, L. Guo, and H. Duan, "A threat assessment model and algorithm based on Elman_Adaboost strong predictor," *Acta Electronica Sinica*, vol. 40, no. 5, pp. 901–906, 2012.

[7] X. Wang, C. Li, and J. Song, "Motion image processing system based on multi core FPGA processor and convolutional neural Network," *Microprocessors and Microsystems*, vol. 82, 2021.

[8] R. A. E. El-Din, O. Marwa, and M. H. El~Din, "Accelerating DNA pairwise sequence alignment using FPGA and a customized convolutional neural network," *Computers and Electrical Engineering*, vol. 92, 2021.

[9] A. Ghani, "Accelerating retinal fundus image classification using artificial neural networks (ANNs) and reconfigurable hardware (FPGA)," *Electronics*, vol. 8, no. 8, 2019.

[10] J.-H. Yi, J. Wang, and G.-G. Wang, "Improved probabilistic neural networks with self-adaptive strategies for transformer fault diagnosis problem," *Advances in Mechanical Engineering*, vol. 8, no. 1, pp. 109–118, 2016.

[11] W. Li, G.-G. Wang, and A. H. Gandomi, "A survey of learning-based intelligent optimization algorithms," *Archives of Computational Methods in Engineering*, vol. 28, pp. 1–19, 2021.

[12] A. Safaei, "System-on-a-Chip (SoC)-Based hardware acceleration for an online sequential extreme learning machine (OS-elm)," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 38, no. 11, pp. 2127–2138, 2019.

[13] A. Hendry and R.-C. Chen, "Automatic License Plate Recognition via sliding-window darknet-YOLO deep learning," *Image and Vision Computing*, vol. 87, pp. 47–56, 2019.

[14] L. Rayson, A. Zanlorensi Luiz, R. Gonçalves Gabriel, T. Eduardo, S. W. Robson, and M. David, "An efficient and layout-independent automatic license plate recognition system based on the YOLO detector," *IET Intelligent Transport Systems*, vol. 15, no. 4, pp. 483–503, 2021.

[15] A. J. Abd El-Maksoud, A. A. Abd El-Kader, B. G. Hassan et al., "FPGA implementation of sound encryption system based on fractional-order chaotic systems," *Microelectronics Journal*, vol. 90, pp. 323–335, 2019.

[16] Z. Cui, F. Xue, and X. Cai, "Detection of malicious code variants based on deep learning," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 7, p. 1, 2018.

[17] Y. Yang, D. Li, and Z. Duan, "Chinese vehicle license plate recognition using kernel-based extreme learning machine with deep convolutional features," *IET Intelligent Transport Systems*, vol. 12, no. 3, pp. 213–219, 2018.

[18] P. Marzuki, A. R. Syafeeza, Y. C. Wong, N. A. Hamid, A. N. Alisa, and M. M. Ibrahim, "A design of license plate recognition system using convolutional neural network," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 9, no. 3, pp. 2196–2204, 2019.

[19] Z. Xu, W. Yang, A. Meng et al., "Towards end-to-end license plate detection and recognition: a large dataset and baseline," in *Proceedings of the European Conference on Computer Vision*, Springer, Munich, Germany, September 2018.

[20] R. Pires de Lima and K. Marfurt, "Convolutional neural network for remote-sensing scene classification: transfer learning analysis," *Remote Sensing*, vol. 12, no. 1, 2019.

[21] Y. Yang, C. Xu, F. Dong, and X. Wang, "A new multi-scale convolutional model based on multiple attention for image classification," *Applied Sciences*, vol. 10, no. 1, 2019.

[22] H. Li, P. Wang, and C. Shen, "Toward end-to-end car license plate detection and recognition with deep neural networks," *Journal of Robotics & Machine Learning*, p. 3813, 2019.

[23] X. Ding, X. Zhang, and N. Ma, "RepVGG: Making VGG-style ConvNets great again," arXiv, 2021.

[24] X. Liu, D. Guo, and L. Cong, "An improvement of activation function in convolutional neural networks," *Journal of Testing Technology*, vol. 33, no. 2, pp. 121–125, 2019.

[25] S. Liu, G. Tian, and Y. Xu, "A novel scene classification model combining ResNet based transfer learning and data augmentation with a filter," *Neurocomputing*, vol. 338, pp. 191–206, 2019.

[26] H. Phan-Xuan, T. Le-Tien, and S. Nguyen-Tan, "FPGA platform applied for facial expression recognition system using convolutional neural networks," *Procedia Computer Science*, vol. 151, pp. 651–658, 2019.

[27] S. Wu, W. Zhai, and Y. Cao, "PixTextGAN: structure aware text image synthesis for license plate recognition," *IET Image Processing*, vol. 13, no. 14, pp. 2744–2752, 2019.

[28] X. Zhou, Y. Gao, C. Li, and C. Yang, "An end-to-end license plate recognition method based on multi-objective optimization and multi-task learning," *Control Theory and Applications*, vol. 38, no. 5, pp. 676–688, 2021.

[29] Y. Feng, S. Deb, G.-G. Wang, and H. Alavi Amir, "Monarch butterfly optimization: a comprehensive review," *Expert Systems with Applications*, vol. 168, 2021.

[30] I. Ghosh and P. K. Roy, "Application of earthworm optimization algorithm for solution of optimal power flow," in *Proceedings of the 2019 International Conference on Opto-Electronics and Applied Optics (Optronix)*, Kolkota, India, March 2019.

[31] J. Li, H. Lei, A. H. Alavi, and G.-G. Wang, "Elephant herding optimization: variants, hybrids, and applications," *Mathematics*, vol. 8, no. 9, p. 1415, 2020.

[32] G.-G. Wang, "Moth search algorithm: a bio-inspired metaheuristic algorithm for global optimization problems," *Memetic Computing*, vol. 10, no. 2, pp. 151–164, 2018.

[33] K. Sirote, S. Apirat, and P. Suttichai, "Multi-Objective optimal power flow problems based on Slime mould algorithm," *Sustainability*, vol. 13, no. 13, p. 7448, 2021.

[34] A. A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, and H. Chen, "Harris hawks optimization: algorithm and applications," *Future Generation Computer Systems*, vol. 97, pp. 849–872, 2019.