Advances in Engineering Research, volume 197
Proceedings of the 2020 9th International Conference on Applied Science, Engineering and Technology (ICASET 2020)

The Study and Analysis of Different CNC Model Based on Image CAPTCHA Recognition

Ligong Cui[1,2,*], Yanyan Cao[1], Liqiang Nie[2]

[1]*College of Mechanism and Electricity, BinZhou Polytechnic, ShanDong Province, China*
[2]*School of Computer Science and Technology, ShanDong University, China*
**Corresponding author. Email: adrainadrain888@gmail.com*

ABSTRACT
With the development of the AI technology, Machine learning and deep learning have been paid attention on increasing extensively. The neural network in deep learning, especially the CNN (Convolutional Neural Network) were used in aspect of image processing and many kinds of AI projects. Deep learning is good at image processing which has resulted in outstanding performance. At the same time, people have difficulties in choosing different CNN model when they use it. This paper picked up a classic project CAPTCHA as a sample. We designed different kinds of CNN model after the studding of LeNet-5 network and VGG GoogleNet etc. First of all, we study the result and the accuracy percent with the same number of CNN layers in different strides, different kernel size and different maps. Then with the rise of the number of hidden CNN layers, two more CNN model were studied with same parameters. We try to find out the relationship between the accuracy percentage and different CNN models. We hope it can do any help to the people who use the CNN model to do some AI project. A kind of the best models was fond in the progress of our studying, it was proved having a good performance not only in the validate accuracy but also in the test accuracy.
Keywords: *TensorFlow, CAPTCHA, CNN, Accuracy, Model Training.*

1. INTRODUCTION

With the development of computer technology and artificial technology, the AI have been using more and more in auto-factory, science study, smart life and smart home etc. we must attach more importance in machine learning especially deep learning. With the CNN have playing an important role in image processing, it makes many ideas come true which we can't imagine and realize or did not do good before, or makes it easier to realize. Many kinds of neural network models such as LeNet, AlexNet, ZF-Net, GoogleNet, VGGNet, ResNet have been used in real projects. Many problems will be met in using the models. For examples, how to choose the structure? how to choose the strides and maps? And what the consequences they will produce on the validation accuracy and test accuracy. There are many kinds of projects in image processing, such as human faces recognition, the plate number recognition and other object recognition. We choose the classic project Image CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart) recognition and choose different structures and different parameters. We compare their validation accuracy after training and try to find out some meaningful reference result to the reader.

We choose the Image CAPTCHA as the study object. Because it is easily to be produced by the computer in Python software environments with t Image-Captcha tool automatically. It has the advantage of produced by random, hardly repetition, large number quantity and convenient. The size of the image CAPTCHA we choose today is 60(height) x 160（width) while its length are 4 bits. The content in CAPTCHA include Arabic numerals, the capital alphabet and the lower alphabet or their mixture as shown in Fig1.

Fig1 The image CAPTCHA

The computer hardware environment includes Conroe i5-8250U CPU, The 1.8GHz Frequency and 8G Memory. We use windows10 operate system as the software environment and other application software including PyCharm, Python3.7, Tensorflow2.0.0rc1 and the third libraries supporting for Python. We take the TensorFlow as the deep learning frame.

2. PREPARE WORK

2.1 The image processing before training

(1) Greying process

The image CAPTCHA was colorful. It has three channels: red, blue and green. In order to be processed by the CNN model easily, the three channels colorful image will be changed in to grey image with one channel. The program sentences are shown as blew.

```
if len(img.shape) > 2:
gray = nump.mean(img, -1)
```

2) The transformation between Text and SVM

Every image CAPTCHA has four characters. Every character possibly will be one of the ten Arabic, twenty-six lower alphabet and twenty-six capital alphabet. We need to transform the text to one-dimensional tensor SVM (support vector machine) with the length of 4 x 62(10+26+26). The one-dimensional tensor SVM need to be transferred to text when the model compares the predict result to the right answer. Then it will know that the predict result is right or wrong.

3) Choose the parameters for the model

In this paper, we set the batch size as 64. We take the Sigmoid cross entropy as the loss function and take the Adam optimizer as the optimizer in Tensor flow. The learning rate is 0.001 and the training number of rounds are 13000 times.

2.2 The realization of CNN model in Tensorflow and Python

We will explain the process and the principle of the CNN model which have the two convolutional layers and two pooling layers. It is shown on Model1.

The size of the image CAPTCHA is 60 x 160 which will import the first convolutional layer. We choose 3 x 3 x 1 as the size of kernel filter, the maps as 32 , the bias quantity as 32,the stride length as 1. We can calculate the parameter number of the first layer from the formula: 3 x 3 x 1 x 32+32=320. The original weight and bias use the initialize result which take Relu as the activate function. The size of the output matrix can be calculated as follow formula when using the SAME padding mode.

$$out_{hight} = \frac{in_{hight}}{stride_{hight}} = \frac{60}{1} = 60$$

$$out_{width} = \frac{in_{width}}{stride_{width}} = \frac{160}{1} = 160$$

The second layer is pooling layer. There are two kinds of pooling method. One is MAX pooling, the other is AVERAGE pooling. We choose the max pooling method here. The kernel size is 2 x 2 and the strides use 2 while the maps doesn't change. So output image size is 30 x 80.

The third convolutional layer takes the kernel size as 3 x 3 x 1,the stride as 1,the maps as 64,the bias as 64 while the input image size is 30 x 80. It's the size of output image of the second layer. So we can calculate the parameter number of the third layer as the formula :3 x 3 x 1 x 64+64=5428.With the SAME padding mode, the size of the output matrix does not change, but maps doubled.

The fourth layer is pooling layer. we also choose the max pooling method here. The kernel size is 2 x 2 and the strides use 2 with the same maps. So output image size is 15 x 40. The program of how to realize the up four layers were shown as follow.

```
#The first layer weight
w_c1 = tf.Variable(w_alpha * tf.random_normal([3, 3, 1, 32]))
#The first layer bias
b_c1 = tf.Variable(b_alpha * tf.random_normal([32]))
#The first convolutional layer
conv1 = tf.nn.relu(tf.nn.bias_add(tf.nn.conv2d(x, w_c1, strides=[1, 1, 1, 1], padding='SAME'), b_c1))
#The first pooling layer
conv1=tf.nn.max_pool(conv1, ksize=[1, 2, 2, 1], strides=[1, 2, 2, 1], padding='SAME')
#The second layer weight
```

```
w_c2 = tf.Variable(w_alpha * tf.random_normal([3, 3, 32, 64]))
#The second layer bias
b_c2 = tf.Variable(b_alpha * tf.random_normal([64]))
#The second convolutional layer
conv2=tf.nn.relu(tf.nn.bias_add(tf.nn.conv2d(conv1, w_c2,strides=[1,1,1,1], padding='SAME'), b_c2))
#The second pooling layer
conv2 = tf.nn.max_pool(conv2, ksize=[1, 2, 2, 1], strides=[1, 2, 2, 1], padding='SAME')
```

The last but the first layer is full connection layer. It will turn the CNN output tensor to a tuple and return a list at the same time. It is calculated by the matrix multiplication and the active function before exported. The last layer is SoftMax. It will export two-dimension vector with the size of 248(4 x 62) after the matrix multiplication.

3. THE EXPERIMENT PROCESS AND THE RESULT ANALYSIS

3.1 The accuracy analysis of different parameters

We designed four different kinds of CNN models with the same convolutional layers, pooling layers and also other layers. But the four kinds of models have different kernel size, different maps and different strides. The four models were shown in Table1.

Table1 Four kinds of different models with same convolutional layers.

Model1

Layer	Type	Maps	Size	Kernel	Stride	Padding	Activation
In	Input	1	60 x 160	—	—	—	—
C1	Convolution	32	60 x 160	3 x 3	1	SAME	ReLU
S2	Max Pooling	32	30 x 80	2 x 2	2	SAME	ReLU
C3	Convolution	64	30 x 80	3 x 3	1	SAME	ReLU
S4	Max Pooling	64	15 x 40	2 x 2	2	SAME	ReLU
F5	Fully Connected	—	1024	—	—	—	ReLU
Out	Fully Connected	—	248	—	—	—	Softmax

Model2

Layer	Type	Maps	Size	Kernel	Stride	Padding	Activation
In	Input	1	60 x 160	—	—	—	—
C1	Convolution	64	30 x 80	3 x 3	2	SAME	ReLU
S2	Max Pooling	64	15 x 40	2 x 2	2	SAME	ReLU
C3	Convolution	128	15 x 40	3 x 3	1	SAME	ReLU
S4	Max Pooling	128	8 x 20	2 x 2	2	SAME	ReLU
F5	Fully Connected	—	1024	—	—	—	ReLU
Out	Fully Connected	—	248	—	—	—	Softmax

Model3

Layer	Type	Maps	Size	Kernel	Stride	Padding	Activation
In	Input	1	60 x 160	—	—	—	—
C1	Convolution	32	58 x 158	3 x 3	1	VALID	ReLU
S2	Max Pooling	32	29 x 79	2 x 2	2	VALID	ReLU
C3	Convolution	64	14 x 39	3 x 3	2	VALID	ReLU
S4	Max Pooling	64	6 x 19	3 x 3	2	VALID	ReLU
F5	Fully Connected	—	1024	—	—	—	ReLU
Out	Fully Connected	—	248	—	—	—	Softmax

Model4

Layer	Type	Maps	Size	Kernel	Stride	Padding	Activation
In	Input	1	60 x 160	—	—	—	—
C1	Convolution	64	58 x 158	3 x 3	1	VALID	ReLU
S2	Max Pooling	64	29 x 79	2 x 2	2	VALID	ReLU
C3	Convolution	128	14 x 39	3 x 3	2	VALID	ReLU
S4	Max Pooling	128	6 x 19	3 x 3	2	VALID	ReLU
F5	Fully Connected	—	1024	—	—	—	ReLU
Out	Fully Connected	—	248	—	—	—	Softmax

The padding method of the model3 and model4 in table1 is VALID. The output image size can be calculated as the following formula. It will use the rounding -off method when the result is not an integer.

$$out_{hight} = \frac{in_{hight} - filter_{hight} + 1}{stride_{hight}} = \frac{60 - 3 + 1}{1}$$

$$= 58$$

$$out_{width} = \frac{in_{width} - filter_{width} + 1}{stride_{width}} = \frac{160 - 3 + 1}{1}$$

$$= 158$$

It will start to training when all the things are ready. We recorded the data of training times and the validation accuracy in the training course. The validation accuracy rate with the training rounds of four kinds of models were drawn in one chart Fig2 after 13000 rounds training.

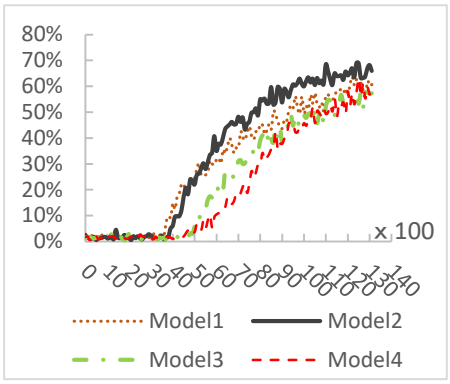

Fig2 Validation accuracy rate of training

Then five times test for accuracy with 100 image CAPTCHAs in one time have been executed to all four kinds of models that we have got after training. We can calculate the average test accuracy in the table. The consequences are shown in the Table2.

Table2 The test accuracy of model1-4

Test / model	Test1	Test2	Test3	Test4	Test5	Average
Model1	18.0%	19.0%	24.0%	23.0%	24.0%	21.6%
Model2	27.0%	27.0%	29.0%	32.0%	28.0%	28.6%
Model3	17.0%	19.0%	13.0%	15.0%	14.0%	15.6%
Model4	19.0%	16.0%	14.0%	16.0%	15.0%	16.0%

It is observed from the Fig2 that the accuracy of model1 and model2 start to rise after 4000 rounds and keep a high trend. Otherwise the left model3 and model4 do not work well as the former two models. It is also verified the same conclusion from the Table2. The test accuracy of model1 and model2 are fairly better than the other two models. The primary reason, I think, is that model 3 and model4 use the greater strides, which make them lose more important characters of the image. Although model4 use even deeper maps, its test accuracy did not improve too much.

On the other hand, mode1 and model2 have different deep maps with 64 or 128, but their performance is better than the latter two models. Model1 did not use the bigger strides in the convolutional layer, so its character extraction is smooth and steady. The best one in the models is model2. The first reason is that it uses the deeper maps. Although it uses even bigger strides at the first convolutional layer not at the second convolutional layer, which makes it loss less character. Model2 has extracted the effective comprehensive characters in the training course and got a better consequence in the test data.

3.2 The accuracy analysis of different layers.

We designed other two kinds of CNN models according to model1. We just add the convolutional layers and pooling layers this time and keep the other parameters unchanged. Table3 Shows the structure of model5 and model6. Model5 add one convolutional layer and one max pooling layer according to model1 while model6 add one convolutional layer and one max pooling layer according to model5. The left parameters keep unchanged.

Table3 Models with different convolutional layers

Model5

Layer	Type	Maps	Size	Kernel	Stride	Padding	Activation
In	Input	1	60 x 160	—	—	—	—
C1	Convolution	32	60 x 160	3 x 3	1	SAME	ReLU
S2	Max Pooling	32	30 x 80	2 x 2	2	SAME	ReLU
C3	Convolution	64	30 x 80	3 x 3	1	SAME	ReLU
S4	Max Pooling	64	15 x 40	2 x 2	2	SAME	ReLU
C5	Convolution	64	15 x 40	3 x 3	1	SAME	ReLU
S6	Max Pooling	64	8 x 20	2 x 2	2	SAME	ReLU
F7	Fully Connected	—	1024	—	—	—	ReLU
Out	Fully Connected	—	248	—	—	—	Softmax

Model6

Layer	Type	Maps	Size	Kernel	Stride	Padding	Activation
In	Input	1	60 x 160	—	—	—	—
C1	Convolution	32	60 x 160	3 x 3	1	SAME	ReLU
S2	Max Pooling	32	30 x 80	2 x 2	2	SAME	ReLU
C3	Convolution	64	30 x 80	3 x 3	1	SAME	ReLU
S4	Max Pooling	64	15 x 40	2 x 2	2	SAME	ReLU
C5	Convolution	128	15 x 40	3 x 3	1	SAME	ReLU
S6	Max Pooling	128	8 x 20	2 x 2	2	SAME	ReLU
C7	Convolution	256	8 x 20	3 x 3	1	SAME	ReLU
S8	Max Pooling	256	4 x 10	2 x 2	2	SAME	ReLU
F9	Fully Connected	—	1024	—	—	—	ReLU
Out	Fully Connected	—	248	—	—	—	Softmax

We got the validation accuracy rate and the losses rate of two kinds of models after 13000 rounds training. The model2 was chosen from above four kinds of models for its better performance. Then the consequences of three kinds of models with different convolutional layers have been drawn in one chart Fig3 and Fig4.

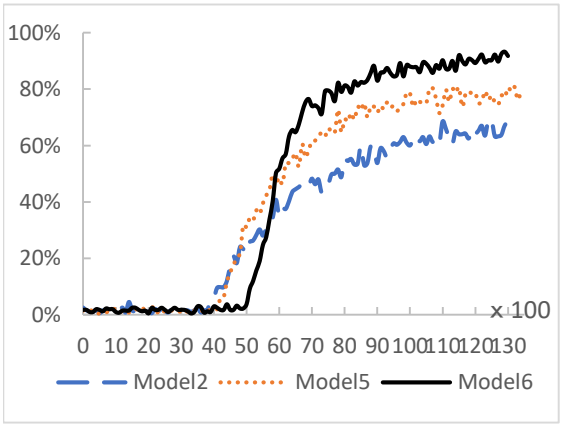

Fig3 The acuuracy with the times of trainning

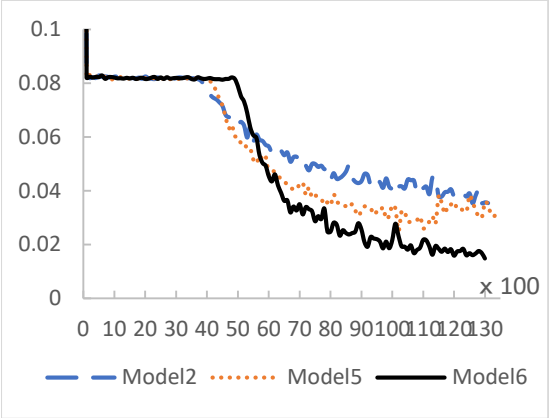

Fig4 The losses rate with the times of trainning

It can be seen from the figure that the validation accuracy and the losses rate have been improved a lot with the increasing of the convolutional layers. Actually, the best validation accuracy of model2, model5 and model6 are respectively 68.25%，80.75%，93.25% and their lowest losses rate are respectively 0.035,0.030,0.014.

Then five times test for accuracy with 100 image CAPTCHAs in one time have been executed to all four kinds of models that we have got after training.

We can calculate their average test accuracy in the table. The consequences are shown as the Table4.

Table4 Test Accuracy of Model2, Model5 and Model 6

test / model	Test1	Test2	Test3	Test4	Test5	Aver	Improv Rate
Model2	27.0%	27.0%	29.0%	32.0%	28.0%	28.6 %	1
Model5	55.0%	56.0%	54.0%	51.0%	55.0%	54.2 %	189.5%
Model6	81.0%	79.0%	84.0%	77.0%	80.0%	80.2 %	280.4%

We can see from the table4 that the accuracy of model5 improved 189.5% compare to the model2 while model6 improved 280.04%. It is easily to find the reason that the maps become deeper, the property of the non-linearization is better, the comprehensive character is easier to extract with the number of convolutional layers increasing. Model6 has the best result of test accuracy.

In practice, we continue to training in model6 to 30000 rounds. Its validation accuracy can reach 99.25% in the final while the losses rate can reach 0.0067. The test accuracy can reach 93%, which is shown at the last row of Fig5. The most error in the test is likely the number 0 and the letter O etc. Error examples are shown in Fig5. We can suppose that the accuracy of the Model6 can be improved further more if the learning rate decreased.

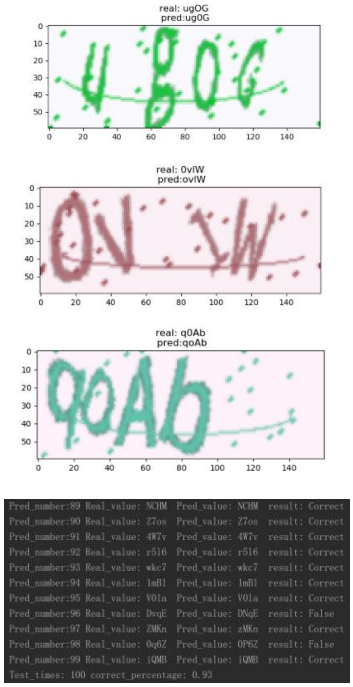

Fig5 The example of error CAPTCHA

4. CONCLUSION

Using the CNN models structure with same layers, it is better to use deeper maps to extract the character of the image CAPTCHA. You should use the bigger strides at the beginning but at last. We advise that you should design a model with more convolutional layers to train when you want to get a better consequence. In order to find better models for deep learning, we will take the learning rate and the batch size into account which we didn't consider this time.

ACKNOWLEDGMENT

This work was supported by Natural Science Foundation of Binzhou city of Shandong province (20191102), and Binzhou Polytechnic foundation Project for Nation Academic visitors.

REFERENCES

[1]ZHANG Zheng, WANG Shunfan, DONG Lei. Captcha Recognition Based on Deep Learning[J].Journal of Hubei University of Technology,2018,33（2）：5-9.

[2]ZHANG Supei，LIU ju，XIAO aowen，DU Zhuang. CAPTCHA Recognition Based on Convolutional Neural

Network[j], Journal of Wuhan Institute of Technology,2019,41(1),89-92.

[3] WU S, ZHONG S, LIU Y. Deep residual learning for image steganalysis［J］. Multimedia Tools and Applications ，2018, 77（9）：10437-10453.

[4] Shi BG, Bai X, Yao C. An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2017, 39(11):2298–2304.

[5] Zhong ZY, Jin LW, Xie ZC. High performance offline handwritten Chinese character recognition using GoogLeNet and directional feature maps. Proceedings of 2015 13th International Conference on Document Analysis and Recognition. Tunis, Tunisia. 2015. 846–850.

[6] P. Xu, Research and application of near-infrared spectroscopy in rapid detection of water pollution, Desalination and Water Treatment, 122(2018)1-4.

[7] Yoo H J. Deep convolution neural networks in computer vision: a review[J]. IEIE Transactions on Smart Processing and Computing，2015，4（1）: 35-43.

[8] P. Xu; N. Na; S. Gao; C. Geng, Determination of sodium alginate in algae by near-infrared spectroscopy, Desalination and Water Treatment, 168(2019)117-122.