Intelligent grading system based on CNN (Oct. 2019)

Ke. Liu,*Automation1703*

***Abstract*—In this paper, a intelligent grading system is proposed. The main work is to segment the picture, recognise a single character, and judge the answers. Both a bp-ANN and a CNN will be tried. And finally, a demo is programed, some ideas on improving the system are proposed.**

***Index Terms*—Image segment, CNN, bp-ANN**

I. Introduction

CNN is widely used in image recognition and Checking children's homework is so boring and time-wasting especially the homework that is easy but without any answer. So an intelligent proposed to handle the problem.

In this paper, an brief online version is realised and an offline version is proposed which is more common use. The online version refers to a process to recognize simple arithmetic written on paper, which means that you input an image of simple arithmetic, the system can recognise the characters and check the answers. And the offline one means that you have to preset the answers and its location, then, what the system does is just recognise the characters and compare with the preset one.

The main parts of this system are image segmentation and character recognition. In order to complete the two parts, orthographic projection and CNN are used. Besides, BP-ANN is also tried in the experiment.

Finally, the outcomes are

1) A trained CNN which can recongnise "0 1 2 3 4 5 6 7 8 9 + - * / ( ) = " and datasets of them made by myself.
2) A gui demo which can do all the work above.
3) Some ideas on improving the project.

II. Related work

Image recognition is not only an introduction project of artificial intelligence, but also an important topic. Especially the handwritten numeral recognition, a lot of related work and dataset are already done and many tools like Tensorflow are availble to build my own CNN easily. On the other hand, image classification is also widely used in many projects, and my project can also be regarded as a image classification problem. But, to finish my project, I have to make a lot more samples of the other seven characters.

2019/10/26
Liuke
Student ID 3170105244
Tel 18888922995

III. Theory

A. Image processing

First of all, the image is preprocessed to reduce noise and made binarized.

Then, segment the image into single character. In this process, orthogonal projections is used. As is show in fig1, first, we projected the image horizontally to get the pixel position, the red part of fig1, which allowed the image to be separated by lines. Then, the image is projected vertically, row by row, to get the pixel position, which allows the image of each row to be separated into columns, thus resulting in a single character.

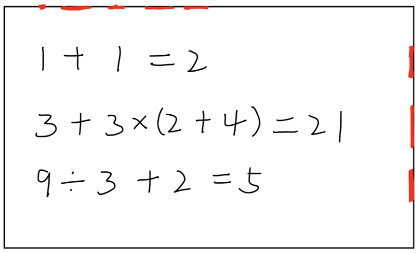


Fig. 1. Orthogonal projections

Next, we get the single characters. When we are doing segmentation, we should also label the pictures, which help to seperate these pictures into different equations and mark its order. Detailed algorithm is: the occurrence of line breaking must be two equations; If the distance between two characters is more than three times the normal distance, the two characters belongs to two equations.

Finally, the image is reshaped as 28*28. The image is composed of pixels, and each pixel point is composed of Red, Green and Blue primary colors, which can be represented by RGB. For example, a 28 by 28 image. The image can be stored as three 28*28 matrices, the first representing the value of R (0-255), the second representing the value of G (0-255), and the third representing the value of B (0-255). In this project, we don't care about color, only black and white colors are needed, and the image above can be converted into a matrix of 28*28, with white to black represented by 0 255. In the actual training, we put into the binary image as a sample.

B. BP-ANN

First of all, I tried BP-ANN to recognise the 17 characters. The structure is shown as fig2. Input is the value of 784 pixels, output is the corresponding encoding, and the activation function is sigmoid.

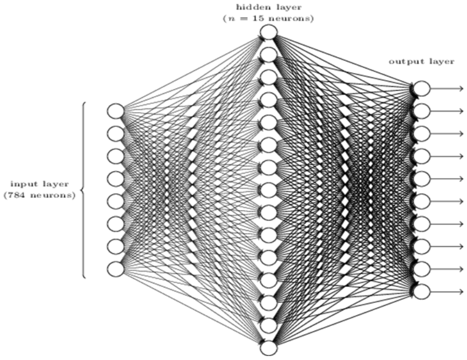


Fig. 2. Structure of BP-ANN

C. CNN

The structure of my CNN is shown as fig3. Three convolution layers, three pooling layers, and a fully connected network are used in this project.

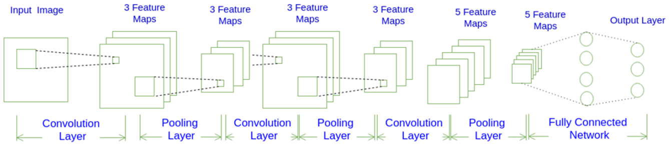


Fig. 3. Structure of CNN

Generally, the function of the convolution layer is to filter the image and extract features, and the function of the pooling layer is to reduce pixels in the image, so as to achieve the purpose of extracting useful pixels and reducing the operation amount. And the full connected layer performs the same function as the BP neural network.

The activation function in CNN is changed into RELU, as fig4

D. Calculate

After we finished recognition, the computer can do the calculating much faster than human. I programed a calculator, but finally, I found that python can simply handle the calculating with the function ”eval()”. And in the image segmentation part, all the images are devided into defferent equations. So it is easy to complete this part.

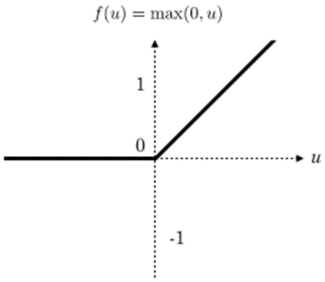


Fig. 4. RELU

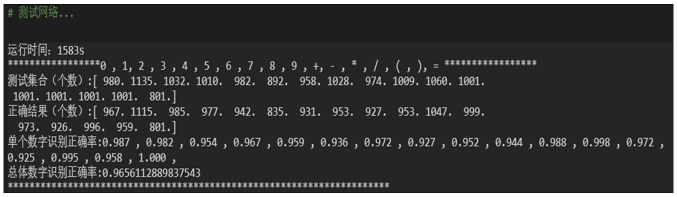
E. Judge

In this part, what need to be done is just comparing the recognised answers with the calculated answers.

IV. Experiment

A. BP-ANN model

BP-ANN is firstly tried in my project. The number of input is 784, the number of hidden neurons is 120, and learningrate is 0.0001. Aftter the ANN is trained, the training accuracy is 97.51%, and the test result is shown as fig5.In this part, the dataset of digit is from MNIST and the dataset of other characters is made by myself.

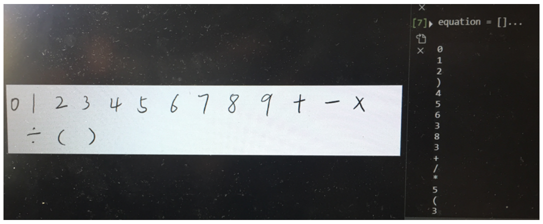


```
# 测试网络...

运行时间: 1583s
****************0 , 1, 2 , 3 , 4 , 5 , 6 , 7 , 8 , 9 , +, - , * , / , ( , ), = ****************
测试集合（个数）:[ 980. 1135. 1032. 1010.  982.  892.  958. 1028.  974. 1009. 1060. 1001.
 1001. 1001. 1001. 1001.  801.]
正确结果（个数）:[ 967. 1115.  985.  977.  942.  835.  931.  953.  927.  953. 1047.  999.
  973.  926.  996.  959.  801.]
单个数字识别正确率:0.987 , 0.982 , 0.954 , 0.967 , 0.959 , 0.936 , 0.972 , 0.927 , 0.952 , 0.944 , 0.988 , 0.998 , 0.972 ,
0.925 , 0.995 , 0.958 , 1.000 ,
总体数字识别正确率:0.9656112889837543
**********************************************************************
```

Fig. 5. Test result

But when I test it with my own handwriting images, this model does not work well as fig6.



```
[7] equation = []...

0
1
2
)
4
5
6
3
8
3
+
/
*
5
(
3
```

Fig. 6. Test with handwriting image

B. CNN model

In this part, CNN model is built with tensorflow, and I made my own dataset of all the characters needed in this project, in order to decrease the training time. There are about 130 images for each character.

The structure of CNN model in this project is shown as fig3. And the details are as follows:

1) All the convolutional images are padded with 0 to ensure the images after convolution are of the same size as the original images.
2) All the pooling layers use max pooling to get the useful pixels. And the kernel is 3*3. Strike is 2.
3) The first convolution layer: 64 3*3*1 convolution kernels, and strike is 1.
4) The second convolution layer: 32 3*3*64 convolution kernels, and strike is 1.
5) The third convolution layer: 16 3*3*32 convolution kernels, and strike is 1.
6) The first and second fully connected layer both consist of 128 neurons.
7) Softmax layer: do a linear regression on the fully connected layer output, calculate the score of each category.

8) Cross entropy is used as loss function.

The model is trained for 2.5×10^4 times. The variety of accuracy and loss are shown as fig7 and fig8. Finally, test the model, the test accuracy is 96.21%

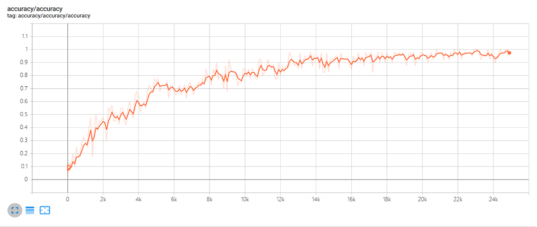


Fig. 7. Accuracy

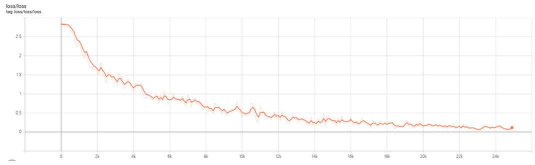


Fig. 8. Loss

C. GUI

I make a gui to run the model and judge the image. The main parts of the gui are image segmentation, recognition, and calculation. The steps to run the gui are:

1) Click the ”Input image” button, choose a image.
2) After the image is shown in the above label. Click the ”Judge” button. You will get a picture shown in the bottom label, the false answers of which will be marked.
3) You can run the gui continuously.

The result is shown below as fig9.

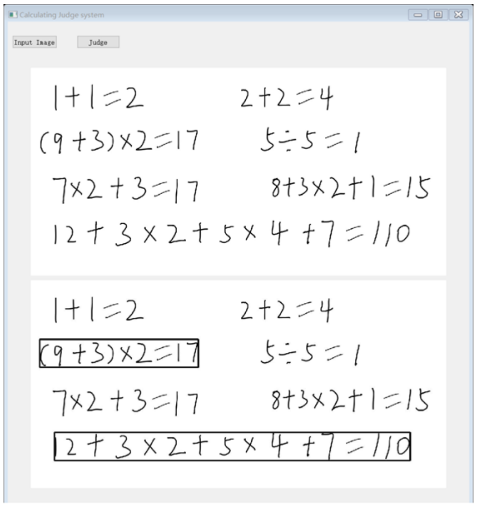


Fig. 9. gui test results

V. Analysis

1) The train and test accurancy BP-ANN is both high. But when I use my own hand-writing images, the model does not work well. In my opinion, the main reason is that the single layer BP-ANN can only reflects individual pixel value, but can not reflects the relationship between pixels. So, if I change the location of the character in image, the character can not be recognised. And if we add more hidden layers, the model will be better.
2) The CNN is much better than BP-ANN. Because the convolution kernel can extract image features, reflect the relationship between pixels and image features at various frequencies. So the location of characters in image will not influence the recognition.

VI. Future development

1) The image processing is not good enough to deal with all the pictures, but only the image with all the charaters separated by columns and the distance equations should be obvious.
2) The dataset is not big enough. So the model is not strong enough.
3) The offline version mentioned above will be more practical in daily life. With specific books and code the pages, we can preset the answers and its location. What the terminal do is recognizing the page by the code, search the answers and locations in database. Then, check it.

VII. Reference

(1) Sun Y, Wang X, Tang X. Deep learning face representation from predicting 10,000 classes[C]//Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on. IEEE, 2014: 1891-1898

(2) https:https://www.tutorialspoint.com/tensorflow/tensorflow_convolutional_neural_networks.htm

(3) https://blog.csdn.net/yunpiao123456/article/details/52437794