Handwritten Chinese Sentence Recognition and Classification Based on Deep Convolutional Neural Networks

Chenyu Wang, Ganquan Wen, Jiali Ge, Runzhou Han, Yuchen Wang
wangcy@bu.edu, wengq@bu.edu, ivydany@bu.edu, rzhan@bu.edu, wangyc95@bu.edu
GitHup: https://github.com/GanquanWen/BU-EC500-Deep-Learing-Project

Figure 1. Handwritten Chinese characters

1. Task

In this project, we are going to study the performance of deep neural network on recognizing the image of sentences written in Chinese characters and give semantic classification on them. The work can be divided into a character recognition model and a sentence classification model. Both of the two models can be achieved with Convolutional Neural Network (CNN). If time allows, we will try to explore the performance of and long short-term memory (LSTM) on longer handwritten sentences classification.

2. Approach

I: Image segmentation

We used Python to extract the characters in an image of a sentence. First, we located the boundary of the sentence. Then we located the boundary of each separate characters.

After binarization, an image can be represented as a matrix consists of 0 and 1. We counted the number of ones in each row and column and sum them up respectively. The sum which is zero can be regarded as the boundary of this character. But many Chinese characters have more than one part. If we segment the character by its boundary, we may divide the entire word into two parts. So we calculated the height of each character by defining the position of ones from top to bottom. Chinese characters can be treated as a square approximately. So if we know the height of each character we can segment it by its width for the reason that each word has approximately same width and height.

II: Character recognition

A deep CNN will be trained on GPUs and will involve Tensorflow library. A configuration from Yuhao Zhang's work can be an ideal reference [1]. In Zhang's paper, he ran most of the experiments on a small number of class of dataset because of the limited GPU power. The performance of different depth is also discussed and 11 layers are discovered to have the best performance. In our work, we may use Zhang's implementation and try to tune it by ourselves.

III: Sentence classification

Based on Yoon Kim's paper, with a simple CNN, we can obtain excellent sentence classification results with only a few parametric adjustments and static vectors [2].

In this part, as characters in the Chinese sentence are not separated, a pretreatment is needed to split the sentences into the characters.

Basically, we plan to do a classification to judge whether the sentence conveys positive emotion or negative emotion. If we have abundant time, we will try to increase the number of classes and divide the sentences into more detailed classes.

3. Completed Tasks

I: HWDB1.1d dataset conversion

The downloaded version of HWDB1.1 dataset which is our handwritten Chinese character dataset is in gnt file which can not be loaded for training. So, we read the gnt file and decode them to png file, both the training set and test set contains 3755 folders for 3755 labels, and the folder name is the label.

II: HWDB1.1 dataset preprocessing

By going over all the images in the dataset, we found some bad images which are just some random points, we delete those images to make sure all the images are valid for training. Also, since the Chinese characters are in different shapes, the shape of images for each character varies a lot, we added some white space to the images either horizontally or vertically depends on the shape of images to make sure the number of rows and columns are equal for each image, by doing so, we will not miss any information in the image and we are able to input them into our deep learning model. Finally, we resized all the images into 100*100 and 58*58, by now, the dataset is valid for training,

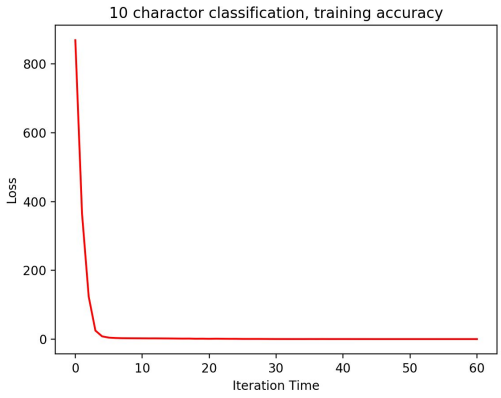
For example, figure 2 is the original image for Chinese character '丁', the number of rows and columns for the image are 63 and 34. We calculated the numerical difference between columns and rows which are 29, and since there are more rows than columns, so we add 2 white spaces whose size are 14*63 and 15*63 vertically to the left and right side of the image, so that the size of new image would be 63 * 63, after resizing to 58*58, we got the new image Figure 3.

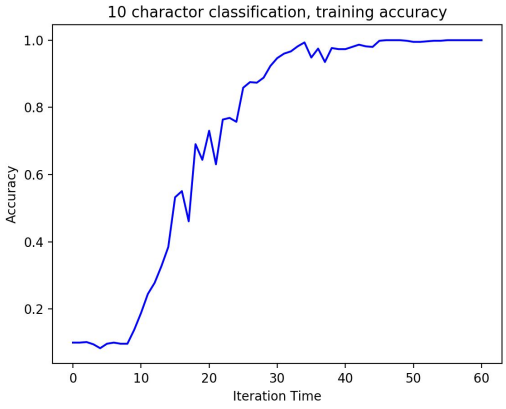
Figure 2. original handwritten Chinese character'丁'

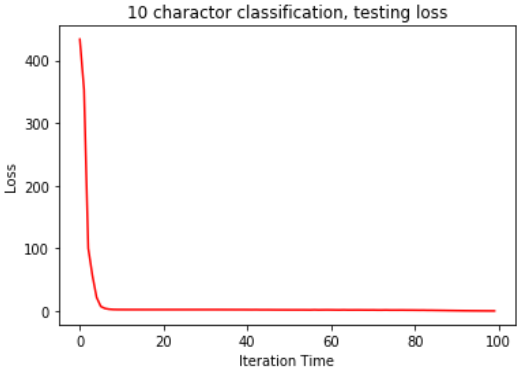
Figure 3. modified handwritten Chinese character'丁'

III.Character recognition

With the preprocessed data, we firstly built a prototype CNN model for a small number of classes. A 10 class discriminator was successfully built. As the number of class is very small, we can train our model without batch. It showed that in the few class case, using the entire dataset in every epoch has high accuracy as well as high efficiency. The following figure shows the loss and accuracy of training set and test set.







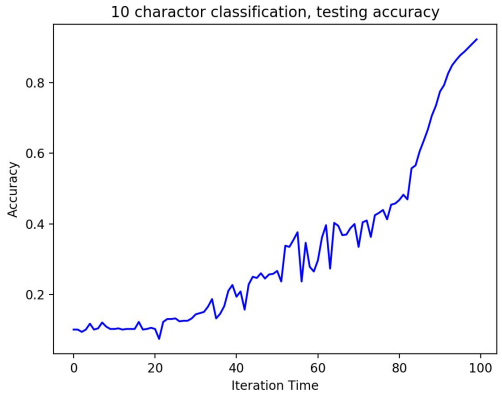


Figure 6. Performance of 10-class discriminator

IV. Image Segmentation

1) Image Preprocess

Using OpenCV to read the image and change it to gray. The scale of gray is from 0 (black) to 255(white). We use 128 as the threshold to binarize the image, under 127 are black dots and over 128 are white dots.

2) Split the Image into Sentences

Scan the image row by row (vertically) and count the amount of dots for each row. We can set a threshold to eliminate the effect of noise.

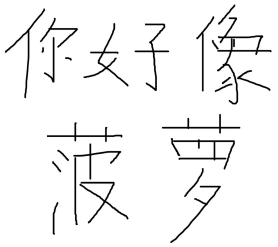


Figure 4. original image which consists of two sentences (5 characters in total)

Then we can get the boundary of each sentence. For example, figure 4 is the original image which contains two sentences. Figure 5 is the vertical scanning of this image. Sentence 1 is located between break point 1 and break point 2. And sentence 2 is between break point 2 and break point 3.

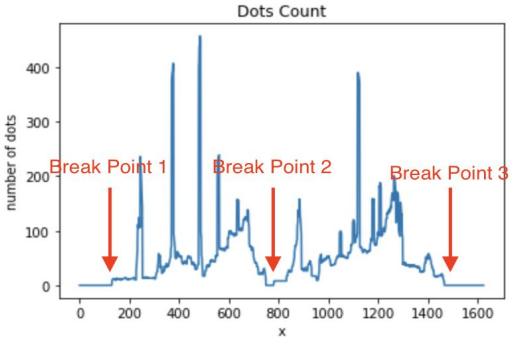


Figure 5. vertical scanning of the example

3) Split Each Sentence into Characters

Scan the sentence column by column (horizontally) and count the amount of dots for each column. It is similar to step 2.

Figure 6. handwritten Chinese character '好'

Also, all the chinese characters are basically a square, we can estimate the size of a single character according to the height of a single sentence. Then we can get the boundary of each character. In this way, we can prevent the program to split a single character in parts by mistake.

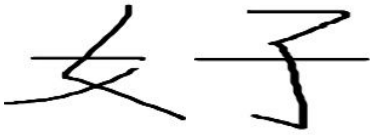


Figure 7. wrong segmentation

For example, figure 6 is the original Chinese character which has two parts. Figure 7 is the example of wrong segmentation which divided this character into two parts.

According to these boundaries, we can locate all the characters and split them. Each character will be put in a single jpg file. Figure 8 shows the final segmentation of the original image(figure 4).

Figure 8. the output of segmentation of the original image

4. Members' Contribution

Task	**Lead**
Preprocess data	Jiali Ge
Image segmentation(extract the characters in an image of a sentence)	Ganquan Wen
Build basic CNN model to train data	Yuchen Wang & Runzhou Han
Adjust parameters of CNN model	Chenyu Wang
Prepare mid-term report	All

5. Future Plan

11.20 -- 11.25:

Improve the accuracy of segmenting images. To make the method be suitable for all kinds of images.

11.20 -- 12.6:

Adjust the parameters of CNN model and increase the number of classes;

Continue to build CNN/RNN model to do sentence classification;

If we have enough time, we will try to build LSTM model for longer handwritten sentences.

12.5 -- 12.10:

Analyze errors and adjust parameters to improve accuracy;

Prepare final report and presentation.

12.12:

Project presentation

6. Challenges

I: Image Segmentation

Sentences and character overlap: If there is overlap between sentences (or characters), the program may see multiple sentences (or characters) as one. This can be solved by adjusting the threshold parameter, but the parameter is decided by how much these sentences (or characters) overlap. So it is hard to find a perfect threshold for this situation.

Punctuation marks: This situation is not covered or be tested by now. This may be challenging because punctuation marks like ", " and "。" are small, if we set the threshold too large, the program may ignore punctuation marks.

Background color: By now, all the test samples have a bright white background and black-colored characters. If the colors of background and characters change, we will need a more complicated algorithm to tell the background and characters apart.

Running time: The image is scanned pixel by pixel, so the running time is totally determined by the size of the image. Resizing the image before scanning can change the running time. But we did not do it. Because we do not know how many characters in an image before we scan it, if the image contains tons of character and we shrink it too much, the characters may be unrecognizable.

II: CNN model

The current CNN model can only give accurate classification on a small number of characters. A huge number of character can cause a lot of trouble, Including how to sample the batch, how to tune the parameters and how to promise the efficiency.

III: RNN model

The connection between our CNN model and our RNN model is a challenge. Because the RNN model classification is based on label of character which are recognized by the CNN model, unifying the the two models' label can be important. However, there are so many characters so the unification can be painful. What's more, there are currently no available 2-class classification model so the training set may not be easy to find. We might change our plan in the next steps in order to take advantage of existing dataset.

References

[1] Zhang Y. *Deep Convolutional Network for Handwritten Chinese Character Recognition*. 2015.
[2] Kim Y. *Convolutional Neural Networks for Sentence Classification*. EMNLP, 2014.