Handwritten Character Recognition

Status Report 2

Patrick Rippl, CM 3353

Brandon Rudolph, CM 3359

Chelsey Yin, CM 2040

Kevin Wang, CM 1996

CSSE – 463 – 01

Project Report #3

Due: Sunday, 02/09/2018

2/9/18

Contents

1 Achievements

The approach depicted in Figure 1 will be implemented in the project and was further improved as shown.

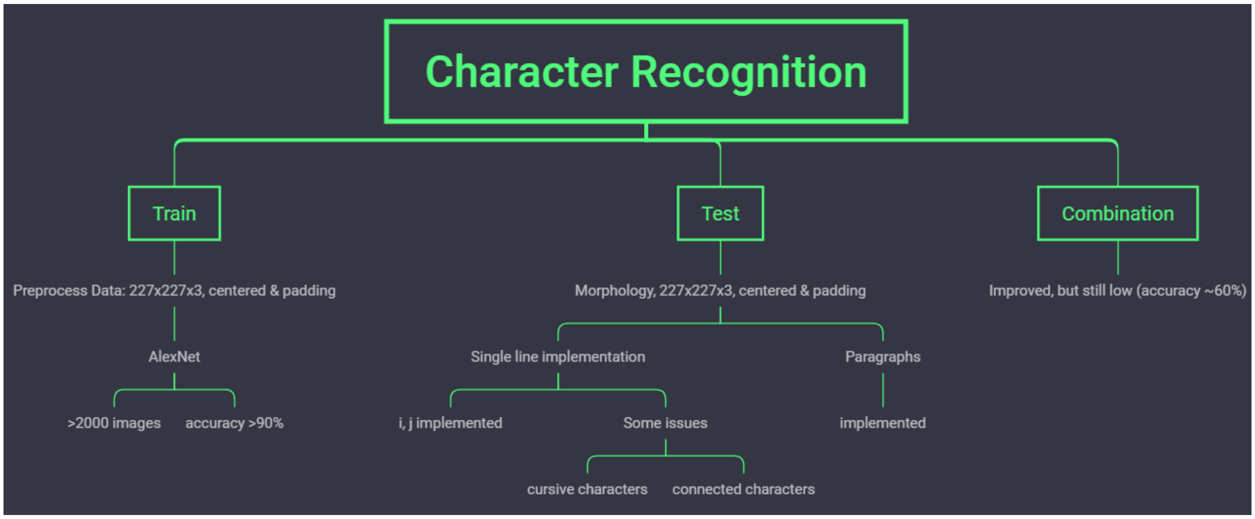
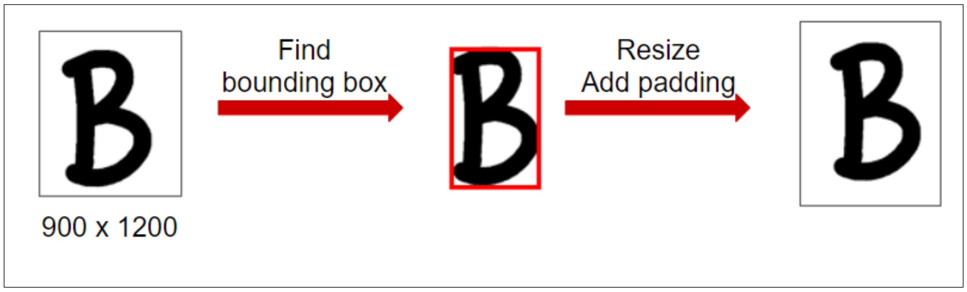


Figure 1 General approach

1.1 Train

The training set we tested led to an accuracy of >90%. Even though the CNN was trained well, the results were disappointing. After discussing, why this happened, a change in the data set was applied. A big change that was made, is the introduction of padding and centering of the characters to be able to train an equal set of features, as well as testing handwritten characters with this net.

The training data contained images with the dimensions 900 x 1200x3. Because AlexNet uses images with the size 227x227x3, the training data was resized by first finding bounding boxes for each character, then resizing that image until one dimension was 227 pixels. After this resizing the images were binarized and padded with whitespace equally distributed to the left/right and upper/lower side, respectively, until the final dimensions were 227x227x1.



\

Figure 2 Process of padding

This approach enabled us to have an equal input of handwritten letters for the AlexNet. After setting the hyperparameters for the net, we got an accuracy of >90% on our training data set again.

1.2 Test

The test set for our CNN will also be preprocessed by the function above. This will enable us to have a consistent dataset. The difficult part in preparing our test data, is the part about separating of letters and morphology operation which will be described further below.

1.3 Combination

After separating handwritten words into single letter, we were able to feed them into our trained AlexNet. The results of this operation were about 60% in accuracy, which is significantly better than it has been before training the network with additional padding.

2 Difficulties

The most difficult part of the work is to prepare our testing data in a way that it can be fed into our trained net.

2.1 Separate Letters in single line words

We can separate each letter in a word, including the "i" s, when there is no cursive writing and the letters are clearly separated. This is a better version than last week, because the data is better preprocessed.

Approach

We checked the overall average position for the x axis of the word. For those pixels that are too far away from the position, we combine the pixels to the nearest component. The threshold for the distance is manually determined.

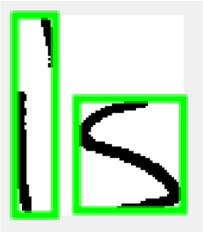


Figure 3 Recognition of pixels within the letters

However, sometimes the dot might be treated on other letters, but the overall number of letters will remain correct.

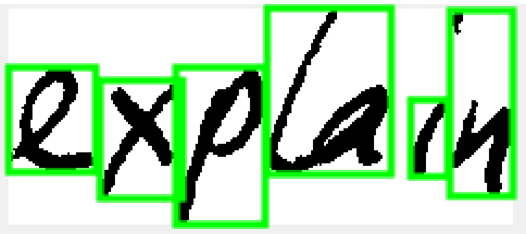


Figure 4 Difficulty in recognition of connected components

Figure 4 shows another problem of our siltation. The combination of ‘l’ and ‘a’ in ‘explain’ is connected and hence we fail to divide them into the corresponding letters.

The next implementation in our algorithm of separation is the separation of lines from a paragraph. We are now able to separate each line in a paragraph as depicted in the following figures.

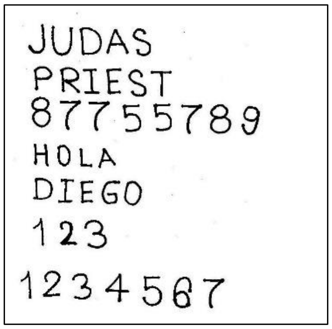


Figure 5 Test data of a paragraph

After separating the lines into single images, we can use the same approach as above to separate the single letters within one word. If there is a large difference between the previous and the current one, we believe it is a new line.

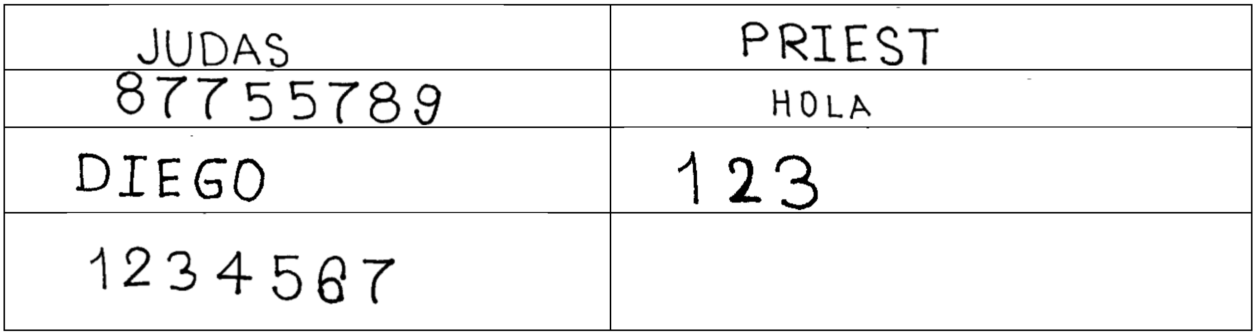


Figure 6 Separation of lines in a paragraph

3 Results and difficulties

Currently, our recognition system can solve the problems we had before in terms of recognizing the letters ‘I’ and ‘j’.

Furthermore, we are not able to separate paragraphs into lines.

It must be stated, that it is difficult to recognize cursive handwriting and handwriting which contains connected letters since they are hard to separate and fit into corresponding bounding boxes.

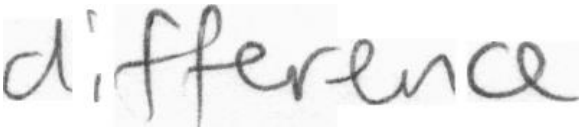


Figure 7 Example data of a handwritten word which is hard to put into bounding boxes

As described above, there is no separation of connected characters.

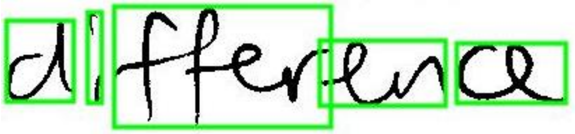


Figure 8 Noise removal and separation of characters

4 Questions

The current threshold to separate letters works for some words but does not work for others. One example of this problem is depicted in the following figures.

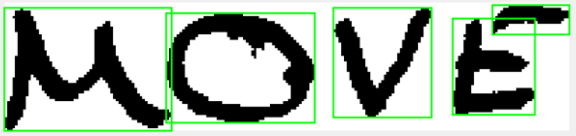


Figure 9 Separation using a specific threshold

With the usage of another threshold, we are able to include the false interpreted pixels to the corresponding letter. However, this method of thresholding does not work for other words. We thus have to make a decision about the level of difficulty in terms of handwritten input we can allow for our algorithm to still be able to work well. This trade-off has to be mentioned before using the handwritten character recognition.

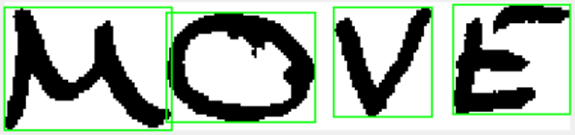


Figure 10 Separation of the same input, using another threshold

After consideration of all the mentioned aspects, we now have to make a decision which handwriting data set is possible to be fed into our algorithm to still be able to recognize most of the data correctly.