Navia Life Care

Internship Selection Assignment

11th January 2021

Omkar Nath

Problem Statement

Captcha Solver

Build a Captcha Solving Solution by any method you wish to choose. e.g. OpenCV, Deep Learning, CNN, OCR etc. Input will a photo of a captcha, and output will be solved text. Captcha is of numeric form. Find the unsolved and solved dataset of captcha in the following link: Give Link Here

Make any assumption you want - that's not an issue.

Captcha datasets are provided with folder “electoral captchas” and “electoral tagged”

Sample Dataset

Following are a few examples of the train and test data.

Train:

Test:

Observations about the Data

A few observations about the data

- Some files such as 88794 and 45431 were missing the png extension.
- Some files were incorrectly named such as 141931.png (14931), 424224.png (42422)
- One test image has just a dot for the fifth digit which is indecipherable – 733.png

Corrections were made wherever these errors were noticed, in the train data folder.

Approach

As we see, there are certain patterns to the captcha, which means that it is not necessary to treat the entire image for analysis such as in a deep learning algorithm. Based on this, the following algorithm was formulated.

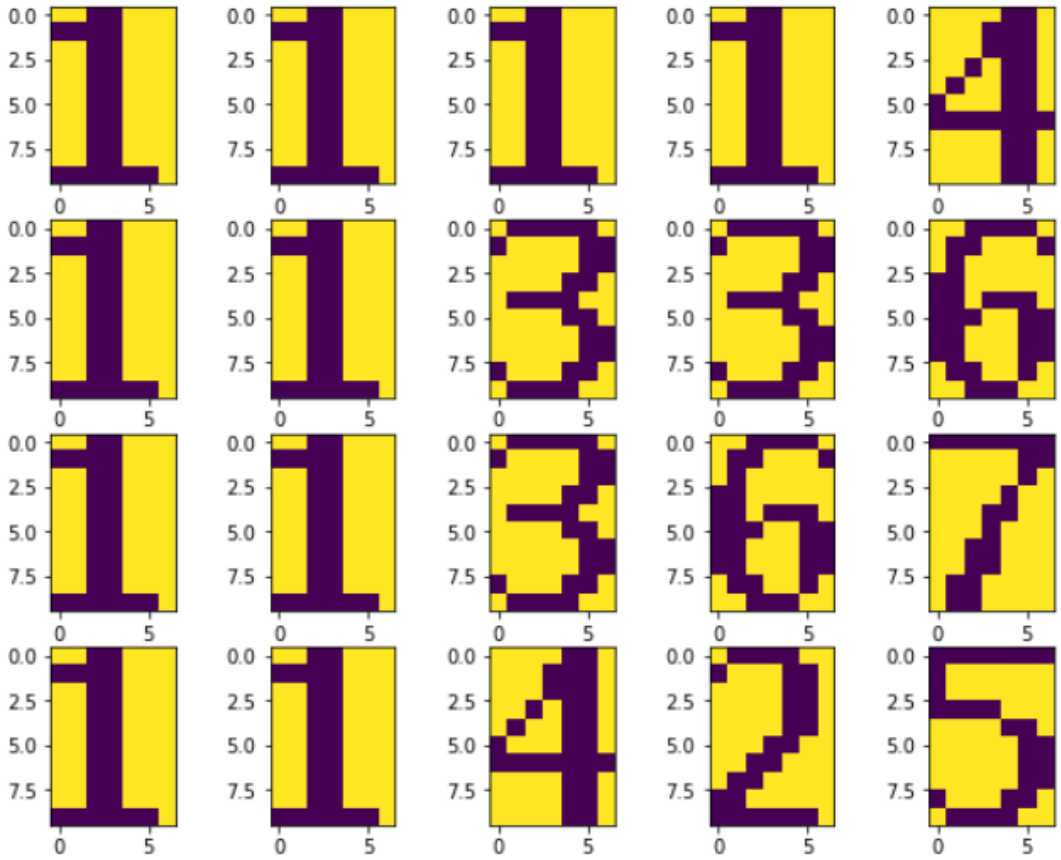
The algorithm takes advantage of a few key observations in the data. These are as follows

- The numbers are all of the same size
- The numbers are all vertically aligned
- There is no overlap between the numbers (horizontal or vertical)
- In a particular vertical pixel column, there is only one digit present at a time
- The same digit is written in almost the same way every time.

All this means that it is possible to linearly separate each digit, and do separate character recognition for each digit. Which is what we do here. Each 130x42 pixel image is broken up into five 7x10 pixel images, each representing one digit. These digits are then identified, and combined to form the original captcha.

For character recognition, SVM is used, which is combined with PCA. This gives results with over 99% accuracy.

Following are a visualization of some of the separated digits.



Algorithm & Codes

1. When loading and dealing with the image file, it is noticed that all the text is simply black, while the background is white. Thus, to simplify the learning, all the pixels with brightness above a certain threshold (red + blue + green values more than 350) are treated as white (assigned 1) while rest are treated as black (0). This binary simplification leads to faster, more accurate training. It also helps to reduce noise (if any).

2. After loading the image, each image needs to be separated into its individual digit images. For this, a simple linear search is done starting from the left. The moment a black pixel is reached, a width of 7 pixels from there is taken. This is done to identify the left and right limits for each digit. Within each of these windows, a similar search is done from the top to identify the vertical limits. This time, a height of 10 pixels is taken. 7 and 10 was decided by analysing various digits, and it was found that all digits can be represented within these.

3. During the above process, two things need to be taken care of.

    a. Firstly, that the black pixel is not just noise. For this, the adjacent pixels are also checked to see if there is some continuity, only then is it considered.

    b. Secondly, the boundaries should not be exceeded (in case some digit goes outside the boundary). In this case. The window is moved slightly inside so as to utilize those pixel values instead.

4. After this, a new data set is created, with 70-pixel values and each having a label corresponding to the digit. This is then used for training. This is done by the first program. A similar test set of all the digits, but without the labels is created by the second program. Instead, the file names are stored.

5. Finally, we come to the third program, which does character recognition. Cumulative variance is used to determine the number of factors for PCA. 8 factors are chosen as this gives a cumulative variance of 1.0. Incremental PCA is thus used.

6. SVM with a rbf kernel (default) is used for making predictions. A K-Folds approach with 3 splits is used for training and finding the optimal parameters. A training accuracy of 99.89% is obtained.

7. The test data is run through the trained model which gives the predictions. These are then merged together for each file, to give the five-digit captcha which is then saved.