**Breaking Captcha**

**Prepared by :**

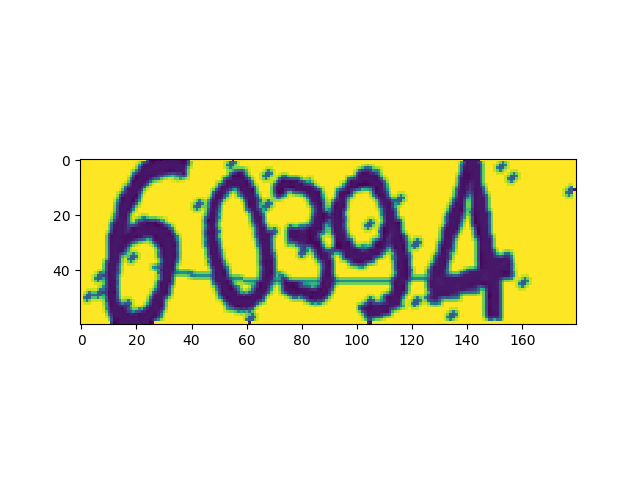
**Ahad Aghapour**

The aim of this project is to breaking image captchas. a program to distinguish human from machine input. in this project we implemented an algorithm in python which is able to recognize the image captchas as well as real targets.

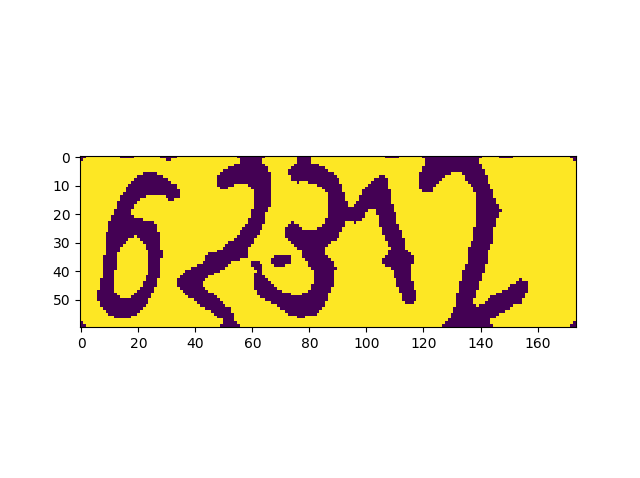
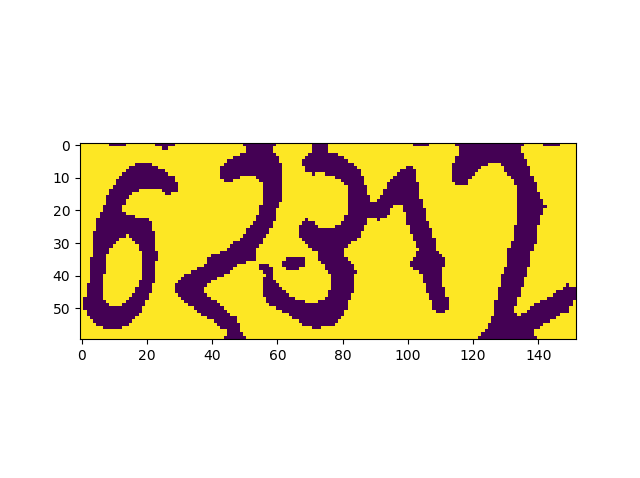
**Approaches:**

Our approach of this is near 91% (for one digit) which can be considered as a good result.

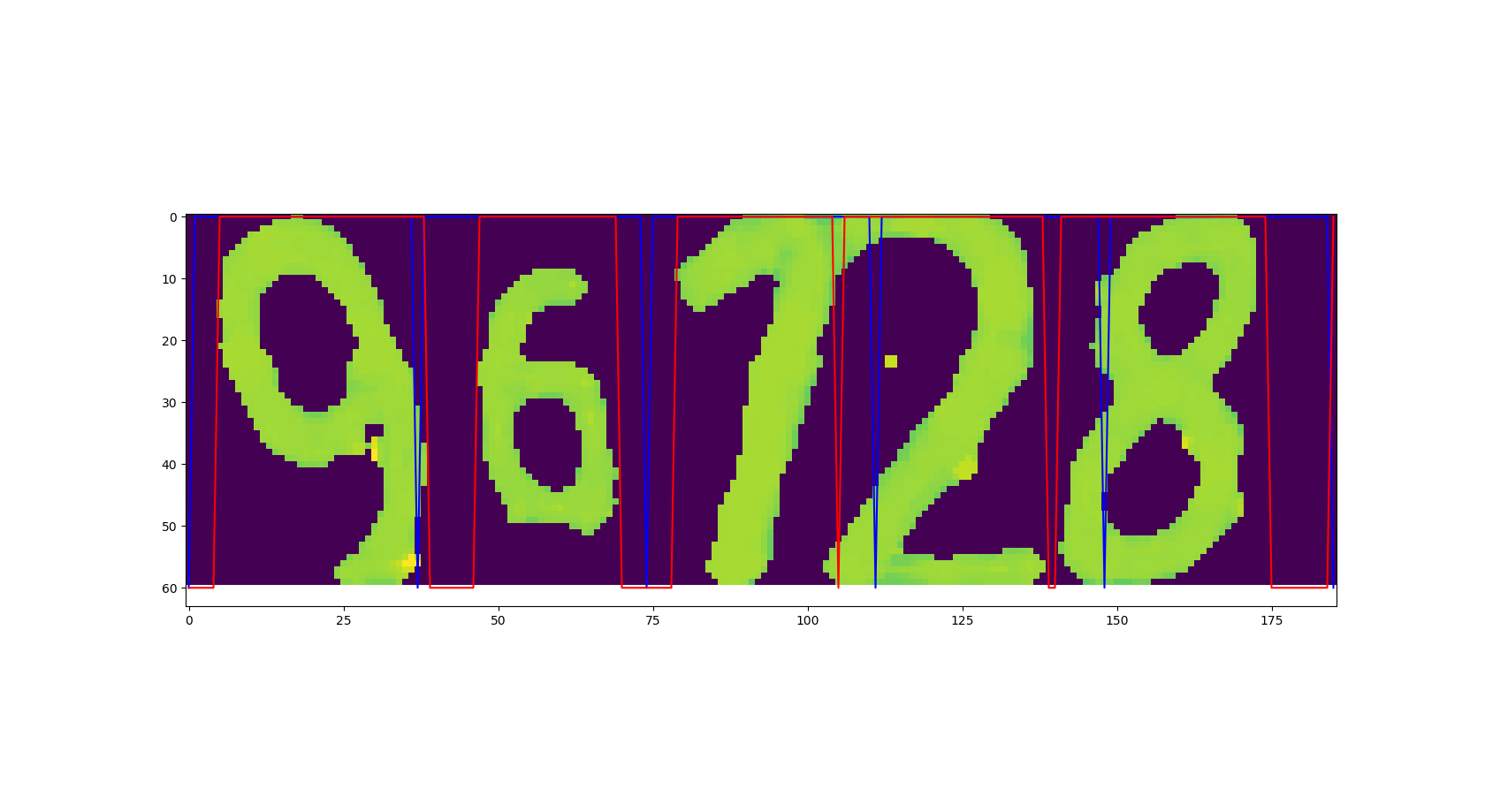
the original image is represented as following(figure 1) sequences of 5 digit numbers inside one captcha. our steps of approaches consist of following :



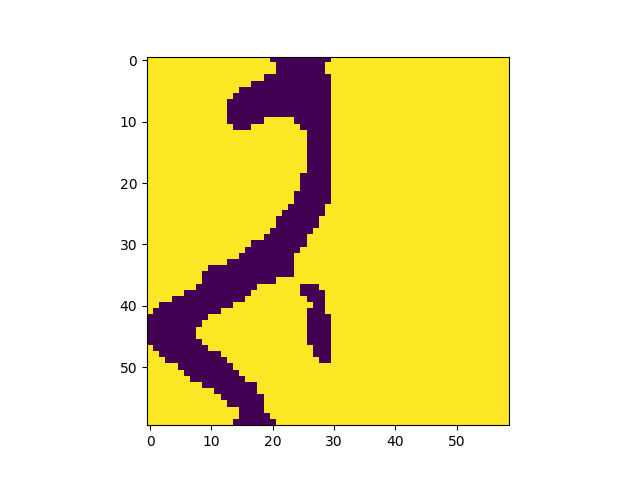
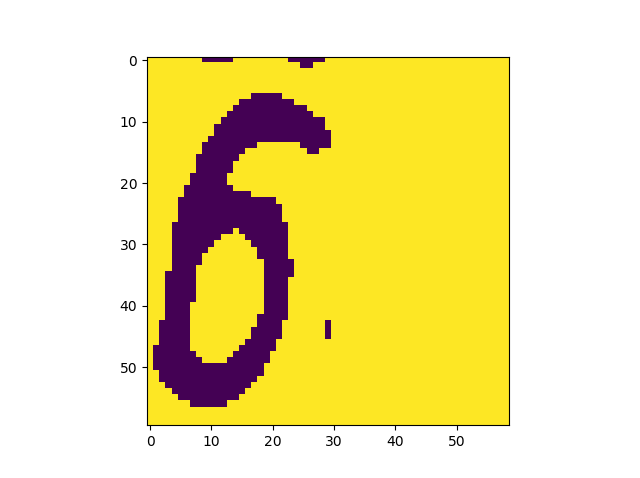
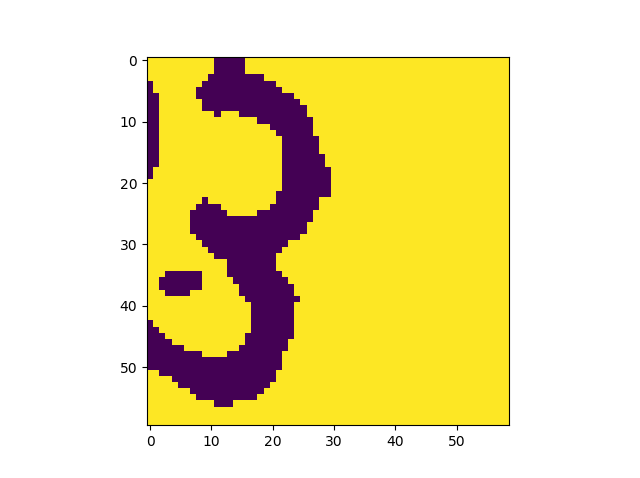
We can change the width of the image by max\_digit variable, (from 1 to n). As Image captcha has 3 channels, first we reduced the image from 3 channel to 1 channel by taking the mean,

Actually instead of using constant threshold 200, we took the mean and use it in our calculations. by this way we separate background and foreground with each other. As raw image has noise(such as dots,lines which are inside the image captcha) it decrease accuracy of the recognition image captcha correctly. Then we use median filter to eliminate noises the code block of median filter is as following : img =scipy.signal.medfilt(imgIn order to achieve better performances the blank spaces in corner of the captcha should be removed we do it by adding columns (each column is set of black and white dots). 

After removing blank spaces and noises from the image, we divide captcha into the equal parts, we have two dividing process first with blue line , second with red line, as following figure the performance of red line is a little better than blue line.



The next step after dividing captcha to equal parts is , for giving image captchas to training we have to do padding on each divided captcha, padding makes the dimensions of image into equal size (row and columns), the final state of the divided image captcha is as following



**Performance evaluation :**

Our evaluate is according to number of max digits, we write a function which named **scoreCalculate** , this method checked that how much the prediction of each digit is successful by observing the max digit value

**The process of running the code:**

**justGetScoreFromPickle**

Forevaluating the model we just need to set justGetScoreFromPickle as true. By changing this parameter as false it fit and train the model and save it pickle file (which is time consuming).

**max\_digit**

We can assign different numbers to this value, it is dynamic and can change during the program.

**Data\_numbr**

In order to building pickle file this parameter helps us to Specify the number of data, which can be 10,000

**test\_number**

Number of test data's

The code block is as following :

# Please set your all parameters from here  
 if \_\_name\_\_ == '\_\_main\_\_':  
 justGetScoreFromPickle = True  
 max\_digs = 5  
 data\_number = 2000  
 test\_number = 2000  
 pickledFile\_name = 'AhadGroup\_prj4.pkl'  
 main(data\_number, test\_number, max\_digs, pickledFile\_name, justGetScoreFromPickle)