ASSIGNMENT 3

Indian Institute of Technology, Kanpur

Department of Computer Science
CS771 Introduction to Machine Learning
Instructor: Purushottam Kar
Submitted: November 28, 2022

M N V RAMACHANDRA ANANTHAPALLI (22111036)
SODANAPALLI LINGA REDDY (22111035)
USHAKIRAN MANNAM (22111063)
YEMIKE ABHILASH CHANDRA (22111071)
GOUNDLA AMITH KUMAR GOUD (22111023)

1 Question 1

ANSWER:

Here we are describing the methods and steps followed to solve the decaptcha problem.

Some Observations given:

The dimensions of each image are known to be 500 * 150 pixels which contains 3 upper case Greek characters

Also given that each image is rotated with only specified angles such as 0, +10, +20, +30, -10, -20, -30 (all angles in degrees)

There are 24 such characters which we need to classify among and this forms a multi-class classification problem

First step is to **pre-process** the images given before feeding them to the Machine Learning algorithm.

Pre-Processing:

1) Identifying and removing background pixels:

Given that Border colour is always a darker version of fill colour of the character. Obfuscating lines in background are of less thickness than the characters themselves.

Preprint. Under review.

Now we need to identify the distribution of the darkness of colours and thickness of lines to remove the background and obfuscating lines.

By using OpenCV, by default the image is read as BGR format. However, this BGR format is less helpful to us as R, G, B in RGB are all co-related to the colour luminance, i.e., We cannot separate colour information from luminance

HSV format of images helps us to separate image luminance from colour information. This helps our task as we can identify luminance of image to remove background.

Now the input image is converted from BGR format to HSV format.
After converting to HSV, a histogram representing the three channels (Hue, Saturation, Value) is plotted to identify the distribution.

Sample input Image is shown below which will be used for explanation purpose:

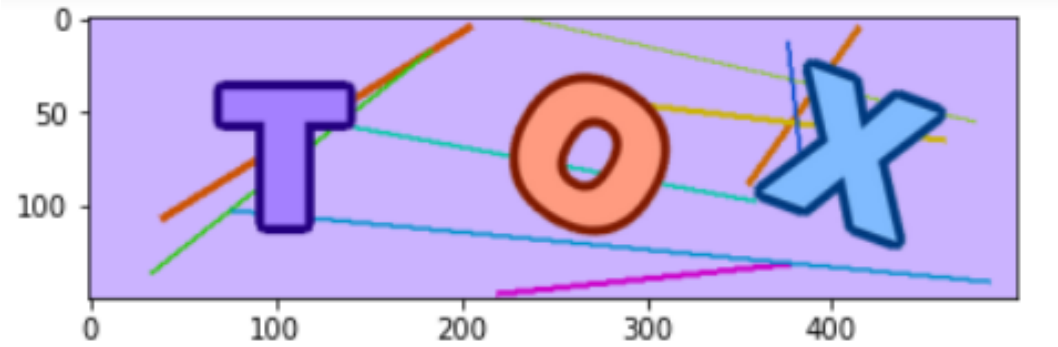

Figure 1: Sample input image

Now histogram for 3 channels is drawn for above image.

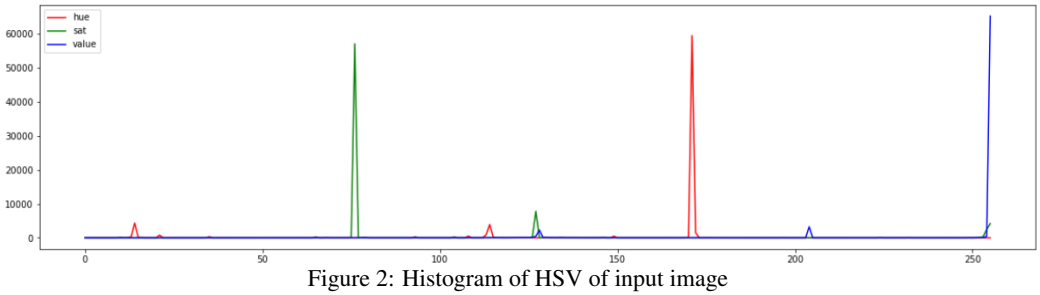

Figure 2: Histogram of HSV of input image

From the Histogram above, the values of saturation can be used to eliminate the colour fill of the alphabets and the background colour. When Sat = 127, then colour fill of the alphabet is removed. When Sat < 127, the background colour information is removed. By using the saturation value background colour and alphabet colour information is removed. Output image is as shown in Figure 3. This method is implemented by modifying the referred code from Reference (1)

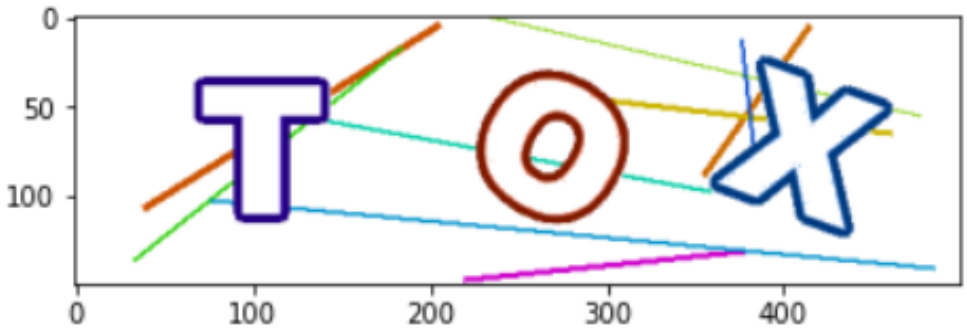

Figure 3: Removed background colour and alphabet colour fill

2) Identifying and removing pixels belonging to Obfuscating lines:

Tried implementing erosion technique, but there are some visible disturbances from the obfuscating lines. So once again by using histogram of the background removed image, we get graph as:

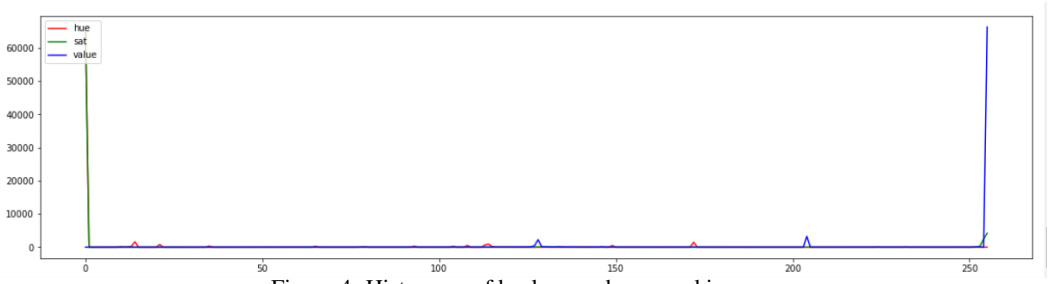

Figure 4: Histogram of background removed image

By removing the pixels belonging to Value channel > 180, all the obfuscating lines are successfully removed as the image given below.

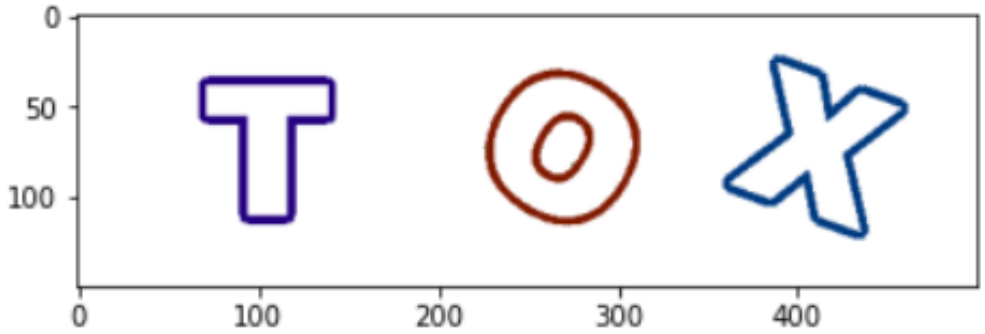

Figure 5: Removed obfuscating lines

3) Segmenting images into pieces each containing of 1 character:

To find out characters present in each image, we used method of calculating sum of vertical column pixels which is as follows:

First image is converted from BGR to Gray Scale. Now we have a single channel instead of 3 channels. Wherever the character is not present will get value as (255) and where there is character,

there will (0) or lesser value. Now we invert the image colour to get (0) pixel value for where the image is not present

Now by using NumPy sum method along axis=0, we get vertical column pixels sum. By this sum we can identify the index of column of start and end position of each character. We record six such indexes, one for starting and one for ending of each character. While segmenting care is taken such that each image will be exactly 10 pixels wide of extra space on both right and left side and 10 pixels are also removed from top and bottom

Implemented by modifying the referred code from Reference (2)
Now the image is segmented using those index values and resulting images are as follows:

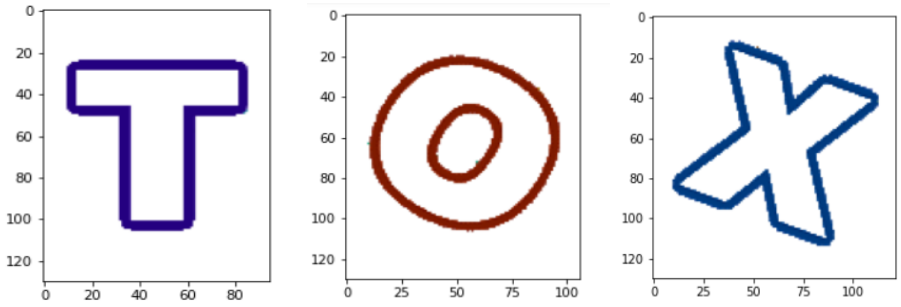

Figure 6: Segmented images with characters separated

4) Removing colour information and scaling the images:

Now as coloured images are of no use in ml algorithm, colour information is removed by converting image from BGR scale to grey scale
First image after converting to grey scale is as follows:

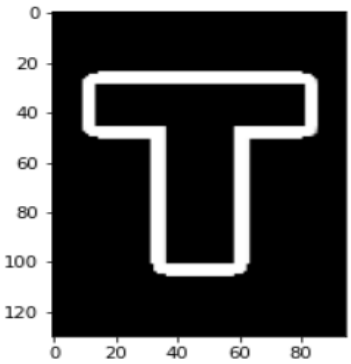

Figure 7: Image after removing colour information

Now each image may be of different sizes and this high-resolution image is not needed for ml algorithm.
So, we scale the image to size of 25*25 pixels.
This results in uniformity of images in sizes.

The scaled image is as follows:

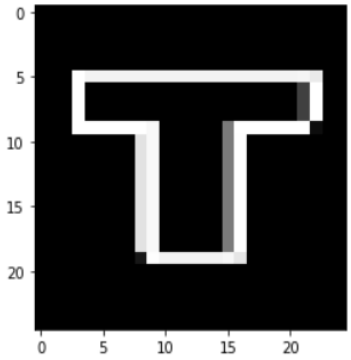

Figure 8: Scaled Image

The pre-processing stage is completed and the images are now ready to be used by ml algorithm

Applying Machine Learning Techniques:

1) Preparing Training data:

From the 2000 training images, all the three characters are segmented which forms 6000 images. They are shuffled randomly.

From the 6000 training images, 10% of them is held for testing purposes and 20% of remaining 90% data is held out for validation phase.

Each image is then converted into NumPy array with the 25 * 25 pixels flattening out into array of 625-pixel values.

Normalisation is done by dividing each pixel value by 255.0

2) ML algorithm:

ML algorithm used is logistic regression with OVR (One vs Rest)
OVR strategy solves multi-class classification problem by breaking down it into multiple binary classification problems.
In our case, since there are 24 alphabets, OVR treats this as 24 binary classification problems.
For example: Alpha vs rest of characters, Beta vs rest of characters.

GridSearchCV is used to find out the best solver among newton-cg, sag, saga and to find out best multi class parameter among multinomial and ovr.

Newton-cg with OVR has given best results among the ones tried with great accuracy.

Model is of size **118 KB** which is low compared to other methods.

This model has given validation and test accuracy of 100%.

For predict.py method, the testing data is similarly pre-processed and the data is formed. This data is fed to model to predict results and the results are split into set of 3 each for each image and returned as comma separated strings.

3) Other ML algo's tried:

SVM with non-linear kernel such as RBF is tried and gives some competitive results. But the main drawback is the size of the model as it went up to 195 MB. So, it is not considered.

SVM with linear kernel is tried, but the model size is still around 2 MB. So, it is also not considered.

CNN algorithm is also tried with 2 Conv2D layers, 2 Maxpooling2D layers and 2 dense layers. This algorithm also gave good results, but the model size is around 1MB. So, this model is also not considered.

References

[1] https://www.freedomvc.com/index.php/2022/01/17/basic-background-remover-with-opencv/

[2] https://nkhandelwal204.medium.com/character-segmentation-using-opencv-and-python-3bf582d9c626