ASSIGNMENT 3

Harshal Rohit, Dheeraj Kumar Pant, Anurag Maithani, Jay Raj Singh, Debdan Bhandari
Indian Institute of Technology Kanpur

Group 11

Abstract

In this assignment we describe a method to perform captcha recognition using image processing and classification techniques. The images were first segregated into individual letters and then classified using Support Vector Machine.

1 SOLUTION

1.1 Dataset

The dataset contained a total of 2000 images each of 600x150 pixels. In each image 3-4 upper-case letters each of size 140x140 were included along with background noise such as abstract lines. The letters had comparatively darker colored border compared to the background. The label of each image consisted the letter names appended in given sequence. The image processing functions used were heavily taken from the opencv library of python called cv2.

1.2 Image Processing

1.2.1 Preprocessing

1. At first, each image was converted into HSV format from given RGB format. The contents of hue(h), saturation(s) and value(v) obtained from the HSV format were separated. Only the "value" part of the image was used for further processing.
2. The value part of the image was passed through erode function with a single iteration. This makes the borders of the letters thiner. Then a GaussianBlur function was used to blur the image. This smoothens the image to reduce noisy background.
3. After the blur function was applied on the value part, it was converted into a binary image by thresholding at threshold value 127(anything above 127 was converted to 255 and rest to 0). This removes most of the background noise which had already become very less prominant due to erosion and blur.
4. For tuning the kernel and sigma values of the gaussianblur function, a set of around 50 images which had the highest noisy background were taken. Then for tuning the kernel, window size were varied from 11x11 to 3x3. For every window size, sigma values were varied from -1 to 3. An optimal value of 5x5 was taken for kernel window and sigma value was taken to be 0.5. The window taken is filled with all ones. The same kernel was was used in erode and dilate functions.
5. Now, the dilate function was applied to thicken back the borders with a single iteration. Due to this, only the borders of the letters were left prominent and the background noise were removed.

1.2.2 Segmentation

1. As the letters consisted of black pixels and the bakground consisted of white pixels, a left to right pass through the image checking the start and end of black pixel sequence would give

the start and end position of each letter. To achieve this, column sums of the pixels were taken. For the places where there was no letter, the sum would be 150x255 = 38250 and wherever there is a letter, the sum would be less than 38250 because it contained some black pixels in place of white. This would also reveal the number of letters present in the image along with the sequence in which they are present.

2. The images thus segregated were resized into 64x64 pixels to make all of them uniform in size. Thus, the training dataset for the classification was created that consisted of 7008 images each consisting of one letter along with the label.

1.3 Classification

For the classification model, support vector machine was chosen and one versus all method was applied. This kept the model size low and prediction time fast.

1.3.1 Training

1. First the images were loaded along with the labels. Three fold cross validation was applied to tune the hyper-parameters of the support vector machine. An optimum value of C was taken to be 5(after searching in the set 0.01,0.1,0.5,1,2,5,10).
2. The images were strectched out into a 1d array of size 64x64 = 4096 values and fed into the classifier for training along with the image labels as Y values. The model was then saved using pickle library for further extraction.

1.3.2 Prediction

1. In the prediction step, The test captcha images were taken and letters were separated using the previous image processing methods used in training. Then SVM model was applied on each letter separately to predict the labels. The labels thus predicted were concatenated into a single string label per image.