PROJECT OCR CSE-573

-Vaibhav Rao (50375332)

Imports =
Just have imported copy (python native module) since python does not allow deep-copy of lists and I was earlier using Binary image for my code.
However I could not make classification work with Binary Images (Since SIFT/ORB etc were having a hard time extracting information) so I switched back to gray scale images and hence just created a copy of the test image.

Below are the things I did for the Enrollment:

For Enrollment, I enrolled the characters provided. I tried multiple things like playing around with Binary images, blurring, edge detection and HOG features. I was not able to get HOG to work due to my kernel crashing everytime during computation, I even tried canny edge detection and gaussian blurring but was not able to get more accuracy.

I Settled for SIFT after resize of our characters. These extracted characters are saved in Res folder however I donot use it since it was a quick process and I am just saving them in a datastructure.

For detection:
I implemented a CCL module for identifying different images and bounding boxes from the test image.
For simplicity of checking if's condition and for thresholding, I converted this to binary.
I run a main loop for M * N pixels in the image. I also Run a recursive function inside which will mark all the points of the first foreground pixel we catch recursively. This helped me in avoiding to run a second-pass to identify connected labels and mark them as 1 label.
Also I have tried to be exhaustive with my code comments so my code is easily understandable.
After that I just Crop all these images and save the BBOX parameters and return the same to
our OCR function.

For Recognition:
I used the parameter extracted from 1st Function (Feature which was a list of SIFT descriptors for our enrolled characters, Our BBOX coordinates from the 2nd function and here I use the Original image (NOT THE BINARY IMAGE) due to issues I faced with SIFT.

I then run SIFT on the cropped images from BBOX. Then I just use a BFMatcher which calculate Norm L2 distances of our test_img and compare
each cropped image with all of our enrolled characters.
Here I select the minimum distance (Sort the distance array(and then compare if the value is less than the threshold.
If the value is less than 290 (threshold – the best threshold after trial and error) I update our Results list with the name as the particular label instead of "unknown"

Results:
Out of the many results as below:

These ones were the best. The F1 score for accuracy was below:

(base) Salonis-MacBook:50375332 saloni$ python evaluate.py --preds results.json --groundtruth data/groundtruth.json

0.5797101449275363

However, I was able to achieve a better accuracy using just SIFT.