Image Processing Report 2

Operating steps:

Firstly, we preprocessed all the images(including train data — class1.jpg, class2.jpg, class3.jpg and also test data — test.jpg). The preprocessing steps are as follows: Firstly do the binary transformation to the image. Then mesh the image to 10\*10 matrix with one channel in which blocks’ value depend on the formula: black pixels over whole pixels in the block. Finally do the binary transformation to the meshed picture. After preprocessing, we find the centroids of the preprocessed pictures. Finally for the classification, we calculate the Euclidean distance between the test sample with each train sample, and then find the classification result with minimum distance.

Data:

Both the train data and the test data are drawn by me on a white board. And since the white board I use has a lot of messy points (noises), I set the threshold of the binary transformation as 48 other than 1. And After meshing and bin-trans, the noise is almost eliminated.

Results:

The test figure is classified to the second class correctly.

Discussion:

Pattern recognition has been a really popular topic nowadays. There are many methods have been developed to get this job done. Some most famous ones are convolutional neural network (CNN), classification and regression tree (CART) and so one. Here I just wanna discuss a optimized method based on the nearest neighbor method — KNN. In the execution of this code, I just use one picture of each class, and clearly we know we can increase the accuracy by using more pictures of each class. While KNN can further improve the accuracy for this clustering problem. KNN’s method is as follows: Choose many pictures for each class and do the preprocessing and calculate each one’s centroid. And then for each test data’s centroid, calculate the Euclidean distance between it and each train data, and find the nearest K (an integer) neighbors to vote its cluster. By doing this, we can ignore some unusual train data’s effect and improve the accuracy a lot.