Project Report

Implementation

First I load all the training images and preprocess them( adjust brightness, resize, and extract SIFT descriptions) then I do the same for the test images. After that, I train the KNN 50\*50 KNN model but do not test it as the instructions say I only have to test two of the models. For the KNN SIFT, I create my data by interleaving the two descriptor lists for the two different-sized pictures and do the same for the labels. I then test the model and print out the True Positive, False Positive, and False Negative Percentages. I repeat that for SVM as well.

NOTE: my program looks for a file called ProjData that contains two folders called Test and Train which each contain 3 folders called bedroom, coast, and forest where the images are stored. The program and ProjData need to be in the same folder for everything to work.

Results:

KNN SIFT Results In Percentage

True Positive: 62.76%

False Positive: 15.08%

False Negative: 22.16%

SVM Results In Percentage

True Positive: 63.60%

False Positive: 16.18%

False Negative: 20.22%

Analysis & Discussion

As we can see SVM has a slightly better correct guess percentage than KNN this is most likely because as your sample size increases KNN loses accuracy to SVM due to potential outliers, while SVM can discard outliers as the sample size grows larger. Since the dimension is relatively high in this scenario SVM gains a slight advantage over KNN here. I found that increasing the iterations from 100 > 200 in SVM actually decreased the accuracy by 2 percent. Similarly halving the iterations from 100 > 50 in SVM significantly decreased the accuracy to a mere 48 percent and cause a large amount of false positives. Interestingly adjusting SVM’s epsilon didn’t seem to change the percentages much if at all. Increasing the K to 3 in KNN lost around 2 percent accuracy and that accuracy loss increases the more Neighbors I included.