HW6 – Part 1: Support Vector Machines for Image Classification

Submitted by: Nafis Neehal

661990881

Part (a): Descriptor Generator:

I generated -

1. Class-wise descriptor for each class in training data
2. Class-wise descriptor for each class in test data
3. Merged all the class-wise descriptor and generated a full descriptor for with data from all classes and custom labels for each class at the last column (both for train and test).

Steps for building the descriptor of the full train/test set –

1. Generate image blocks of size b_h x b_w
2. Calculate all the pixel blocks - merge all three channels into one continuous block with 3 columns side-by-side, each row for one pixel, each column - 1 channel
3. Generate histograms for all the block using "histogramdd"
4. Flatten each of the histogram for each block
5. Merge all the histograms of the full image into a single vector
6. Generate the final descriptor of one single image with after concatenating class label as the last column (1x1025)
7. Now this is done for all image in each of the class
8. Finally, each of the class descriptors are merged and generate the final descriptor of all the classes with class label.
9. Using Python's Pickle model dumped the descriptor for train and test.

Class Labels –

- Grass = 1
- Ocean = 2
- Redcarpet = 3
- Road = 4
- Wheatfield = 5

Part (b): Support Vector Machine for Image Classification:

Kernel: Linear (using LinearSVC)

Steps for training:

1. Load the .pkl file containing descriptor of all the training data
2. Load all the data into program shuffle and normalize.
3. Run a Grid Search over a range of C values using GridSearchCV(). 5-Fold cross validation was performed.
   Range of C is = np.linspace(0.1,10,100, endpoint=True) = [0.1, 0.2, 0.3, …, 9.8, 9.9, 10.0]
4. Obtain 5 different classifiers each initialized with a value of C which best classifies only one class (1 versus Rest classifiers)
5. Get the Weights and Biases for each of the classifiers. Normalize the Weights. This will be used as decision criteria for class assignment to each image while predicting.
6. Train all the data with each of the classifier, add bias, and then divide by the norm of the weight vector.
7. Then predict labels for all inputs. Calculate training error for each class by comparing the predicted label for training data with the original label for the training data.

Steps for testing:

1. Import the .pkl file with descriptor of test files
2. Shuffle and normalize the test data (same as training data)
3. Predict label for test data according to the decision schema given in HW Statement
4. Calculate classwise accuracy.

Performance Analysis:

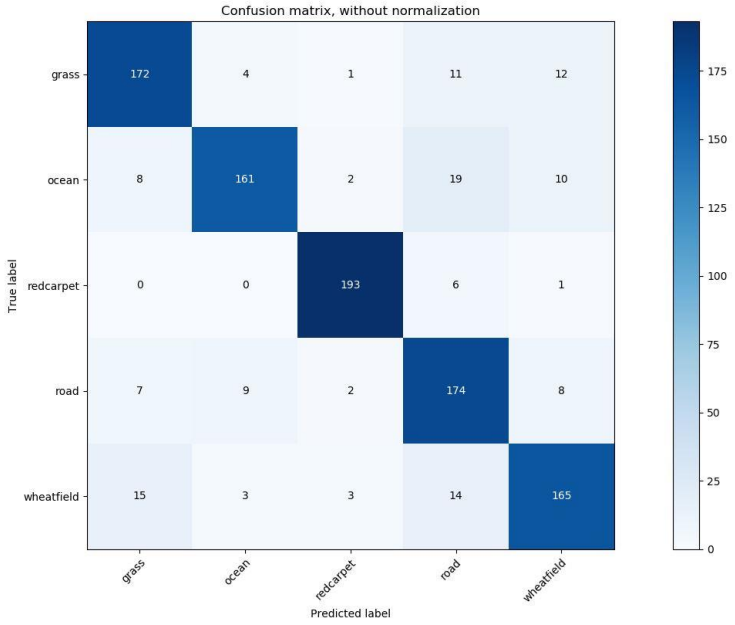
Some of the trial-and-error values that I tried (without using GridSearchCV):

C	Max_iter	Training Error	Test Accuracy
0.001	1000	[0.095, 0.03625, 0.00125, 0.3125, 0.1825]	[0.665, 0.75, 0.94, 0.46, 0.645]
0.01	1000	[0.26125, 0.065, 0.00125, 0.085, 0.1175]	[0.54, 0.7, 0.935, 0.645, 0.68]
0.5	1000	[0.2625, 0.065, 0.00125, 0.085, 0.1175]	[0.54, 0.7, 0.935, 0.645, 0.68]
1	1000	[0.2625, 0.065, 0.00125, 0.085, 0.1175]	[0.54, 0.7, 0.935, 0.645, 0.68]
5	1000	[0.2625, 0.065, 0.00125, 0.085, 0.1175]	[0.54, 0.7, 0.935, 0.645, 0.68]
50	1000	[0.2625, 0.065, 0.00125, 0.085, 0.1175]	[0.54, 0.7, 0.935, 0.645, 0.68]
100	1000	[0.2625, 0.065, 0.00125, 0.085, 0.1175]	[0.54, 0.7, 0.935, 0.645, 0.68]
0.01	2000	[0.12125, 0.105, 0.00125, 0.02875, 0.2475]	[0.67, 0.645, 0.93, 0.705, 0.545]
0.1	2000	[0.12125, 0.105, 0.00125, 0.02875, 0.2475]	[0.67, 0.645, 0.93, 0.705, 0.545]
1	2000	[0.12125, 0.105, 0.00125, 0.02875, 0.2475]	[0.67, 0.645, 0.93, 0.705, 0.545]
10	2000	[0.12125, 0.105, 0.00125, 0.02875, 0.2475]	[0.67, 0.645, 0.93, 0.705, 0.545]
100	2000	[0.12125, 0.105, 0.00125, 0.02875, 0.2475]	[0.67, 0.645, 0.93, 0.705, 0.545]
1	3000	[0.19625, 0.085, 0.00125, 0.14, 0.08375]	[0.575, 0.645, 0.93, 0.555, 0.665]
1	4000	[0.12625, 0.09, 0.0025, 0.04125, 0.22875]	[0.635, 0.64, 0.925, 0.68, 0.56]
1	10000	[0.12625, 0.1275, 0.0, 0.14125, 0.14]	[0.6, 0.58, 0.93, 0.58, 0.585]
0.001	20000	[0.185, 0.0825, 0.0025, 0.13625, 0.1875]	[0.585, 0.65, 0.925, 0.57, 0.51]
0.1	30000	[0.21375, 0.14, 0.00375, 0.105, 0.18]	[0.555, 0.555, 0.93, 0.58, 0.535]
10	50000	[0.12, 0.13125, 0.00375, 0.11, 0.375]	[0.675, 0.575, 0.93, 0.585, 0.405]
0.01	100000	[0.1975, 0.1425, 0.00375, 0.14375, 0.26375]	[0.53, 0.595, 0.93, 0.555, 0.495]
1	100000	[0.2875, 0.18125, 0.00375, 0.0875, 0.32125]	[0.505, 0.575, 0.93, 0.615, 0.45]

Final best value of C and training error and test accuracy for different classes using GridSearchCV():

Class	**C - Value**	**Training Error**	**Test Accuracy**
Grass	0.9	0.140	0.860
Ocean	0.4	0.195	0.805
Red carpet	1.1	0.035	0.965
Road	0.8	0.130	0.870
Wheatfield	0.7	0.175	0.825

Confusion Matrix:



Precision, Recall, F1-Score and Overall Accuracy: