

Image classification through Visual Bag of Words

Report

Bag of words is a popular classification technique used for text classification and now is also used for image classification where an image is described in terms of patches or visual words. These patches or visual words are the key points in an image detected through a key point descriptor such as Sift by David Lowe.

A vocabulary is built from these descriptors which are 128 dimensional each and detected from every training image through Sift with max 50 descriptors, stored in a list and then clustered through a clustering algorithm like Kmeans, into k optimal clusters. The centers of each cluster represents a visual word in vocabulary. This process does not involve any spatial information of the key points thus is an order less representation of image key points but it still has achieved 50-60% accuracy.

Test images are classified based on this visual vocabulary through a classifier. For comparison two different classifier performance is compared, KNN and linear SVM. The input given to these classifiers are histograms that represent an image through the vocabulary, i.e how many times each visual word was present in a particular image. For this process, key points are detected from an image through Sift key point detector with max descriptors to be 250, many values were used for testing purposes but 250 was optimal. Each descriptor in a particular image is compared with the visual words by calculating Euclidean distance with every visual word, the word with least distance gets a vote in histogram at the same location[i] where it was in vocabulary.

Both these processes are time and resource consuming especially when vocabulary size is large or the maximum sift descriptor are very high. Therefore a sample of descriptors is used in both the process but different sample count and both the vocabulary and image features are saved through pickle for further use and easy testing.













Training data had total 15 categories whereas test data had only 4 categories and the size of test images was 705 and 1500 training images.

KNN Classifier

First classifier to test on is KNN. Euclidean distance metric is used for similarity measure between the train and test image features which are of shape 1500xvocab_size and 705xvocab_size respectively. Performance is measured for various vocab sizes like 10, 50, 100, 200, 250, 300 and 400 but 200 gave better accuracy than other vocab sizes. The reason behind this is that too many clusters will have very less intra cluster distances as a data point belonging to both the clusters will suffer. Too less clusters will not be able to represent the data well.

Similarity of a test image is calculated with every train image, distances being saved, and then with the help of np.argsort, the array containing distances is sorted ascendingly and returned an array of the indices of sorted array. The first k indices are selected, converted into a dictionary where keys are the labels/categories accessed from the train labels and values being the number of time each category is present among the k neighbors. The category with highest count is the predicted label for the test image. Many values for k were tested like 3, 5, 7, 9, 11, 13, 15, 17, 19 and 21 but k=15 gave the highest accuracy.

This process is repeated for every test image. The various results along with the confusion matrix, accuracy score, max features detected through Sift are as follows. The visualization of the test categories as TP, FN, FP and sample image as per the highest accuracy model with vocab size 200 and k=15 is as follows

Categories	Sample training image	Sample True Positive	False Positive with true label	False negative with wrong label
Bedroom			<i>Coast</i> 	<i>Store</i> 
Coast			<i>Bedroom</i> 	<i>Tall Building</i> 
Forest			<i>Bedroom</i> 	<i>Suburb</i> 



KNN with vocab size = 10, k=7, and sampling 1/3, max descriptors=200

Accuracy = 39%

Category	TP	FP	TN	FN	
0	Kitchen		0	4	701
1	Store		0	10	695
2	Bedroom		9	8	581
3	LivingRoom		0	29	676
4	Office		0	4	701
5	Industrial		0	6	699
6	Suburb		0	64	641
7	InsideCity		0	10	695
8	TallBuilding		0	11	694
9	Street		0	25	680
10	Highway		25	44	560
11	OpenCountry		0	9	696
12	Coast		154	40	405
13	Mountain		0	30	675
14	Forest		193	30	447

Confusion matrix, without normalization

True label

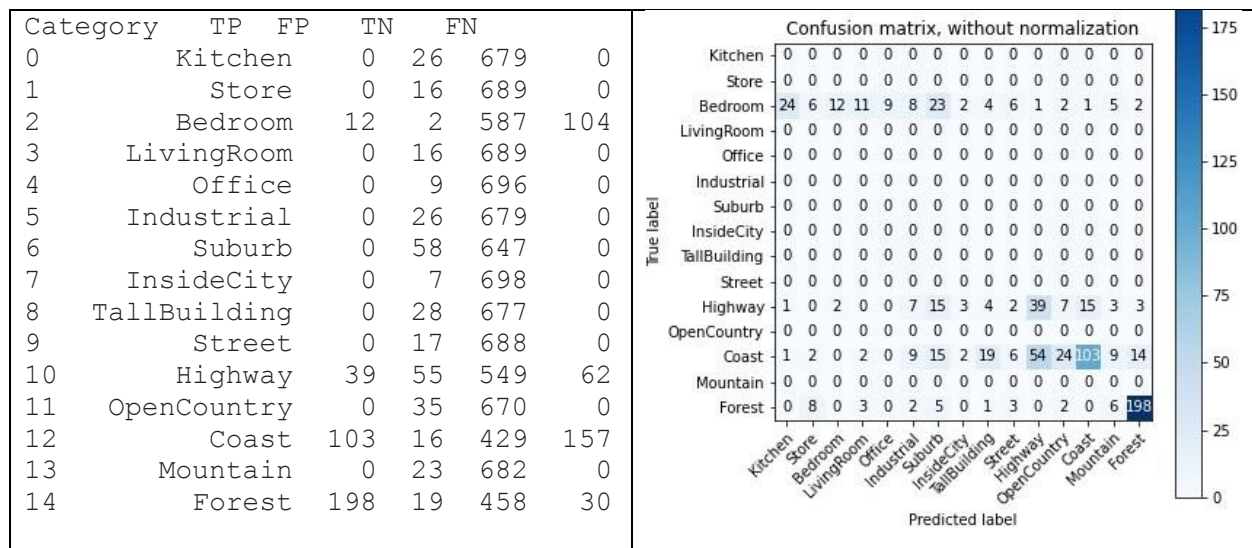
Kitchen	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Store	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bedroom	17	7	7	20	3	8	14	5	4	9	4	4	1	6	7				
LivingRoom	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Office	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Industrial	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Suburb	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
InsideCity	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TallBuilding	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Street	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Highway	3	1	4	4	2	6	6	9	7	2	20	7	23	4	3				
OpenCountry	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Coast	4	1	4	4	8	12	9	13	19	25	40	14	8	16	7				
Mountain	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Forest	3	18	2	6	2	3	15	1	2	3	0	6	0	3	6				

Predicted label

Kitchen	Store	Bedroom	LivingRoom	Office	Industrial	Suburb	InsideCity	TallBuilding	Street	Highway	OpenCountry	Coast	Mountain	Forest
---------	-------	---------	------------	--------	------------	--------	------------	--------------	--------	---------	-------------	-------	----------	--------

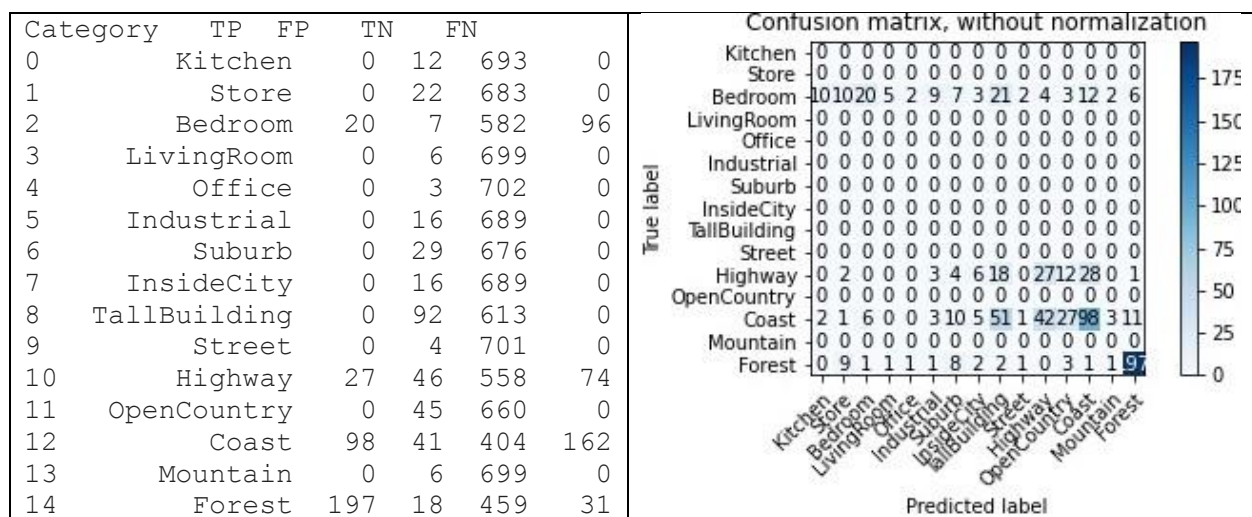
KNN with vocab size = 50 , k=13, and no sampling of descriptors, max=50, 250

Accuracy = 50%



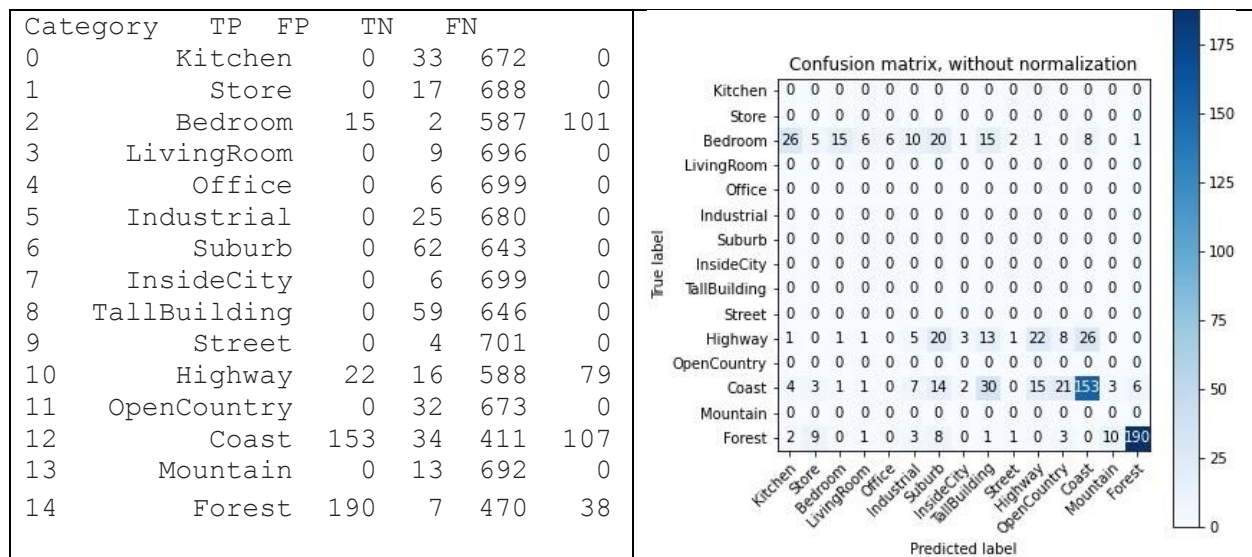
KNN with vocab size = 100 , k=11, and sampling 1/3 of descriptors, max=200

Accuracy = 48%



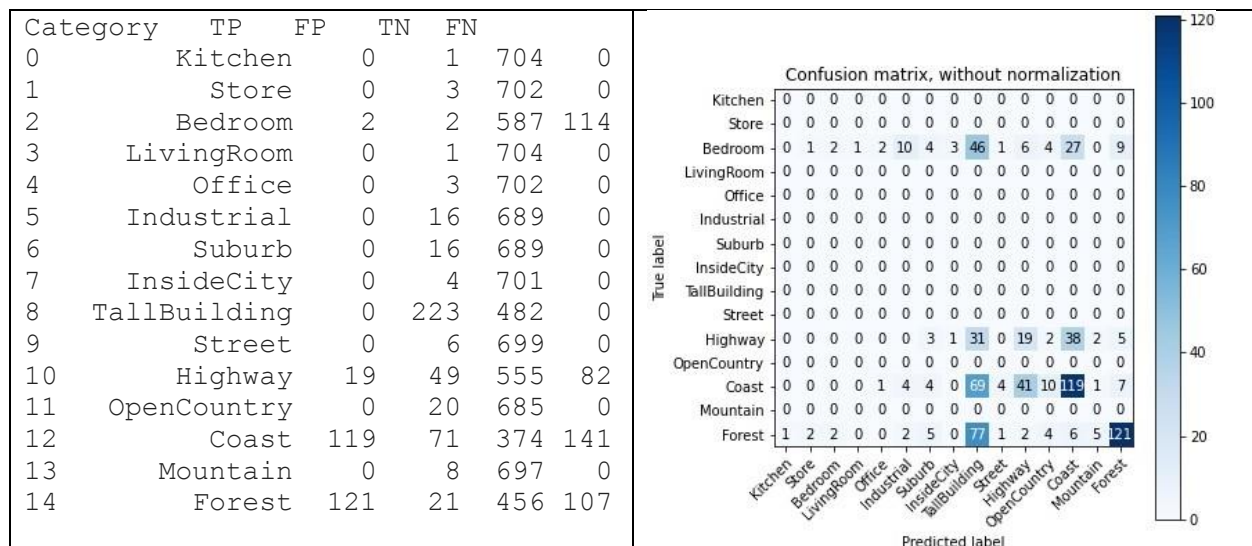
KNN with vocab size = 200, k=15, descriptors max=50, 250

Accuracy = 54%Highest accuracy among the experiments performed












KNN with vocab size = 400 , k=5, and sampling 1/3 of descriptors, max=200 and

Accuracy = 37%



SVM Classifier

As per the instructions and the starter code, Linear svm was allowed to use. The Sklearn.LinearSVC is a one vs all linear svm classifier that generated 15 classifiers according to our data. The regularization parameters like random state and tol were tuned to achieve better performance. SVM model was fitted for various vocab sizes i.e 10, 50, 100, 200, 250, 300, 400 and the results predicted with the built in .predict method. The best performance was achieved for vocab size 10.

Categories	Sample training image	Sample True Positive	False Positive with true label	False negative with wrong label
Bedroom			<i>Coast</i> 	<i>Livingroom</i> 
Coast			<i>Bedroom</i> 	<i>Forest</i> 
Forest			<i>Bedroom</i> 	<i>Mountain</i> 
Highway			<i>Bedroom</i> 	<i>Coast</i> 

SVM with vocab size = 10 and descriptors max=200, random_state=2, tol=1e-05

Accuracy = 54%

Category	TP	FP	TN	FN	
0	Kitchen	0	4	701	0
1	Store	0	10	695	0
2	Bedroom	9	8	581	107
3	LivingRoom	0	29	676	0
4	Office	0	4	701	0
5	Industrial	0	6	699	0
6	Suburb	0	64	641	0
7	InsideCity	0	10	695	0
8	TallBuilding	0	11	694	0
9	Street	0	25	680	0
10	Highway	25	44	560	76
11	OpenCountry	0	9	696	0
12	Coast	154	40	405	106
13	Mountain	0	30	675	0
14	Forest	193	30	447	35

Confusion matrix, without normalization																			
True label	Kitchen	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Store	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Bedroom	2	6	9	22	2	1	26	1	2	6	11	5	7	4	12			
	LivingRoom	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Office	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Industrial	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Suburb	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	InsideCity	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	TallBuilding	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Street	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Highway	1	0	3	3	1	2	12	2	2	4	25	2	31	9	4			
	OpenCountry	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Coast	1	0	4	1	0	2	17	6	5	11	31	2	5	12	14			
	Mountain	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Forest	0	4	1	3	1	1	9	1	2	4	2	0	2	5	9			
Predicted label																			
	Kitchen	Store	Bedroom	LivingRoom	Office	Industrial	Suburb	InsideCity	TallBuilding	Street	Highway	OpenCountry	Coast	Mountain	Forest				

SVM with vocab size = 50 and descriptors max=50,250

Accuracy = 54%

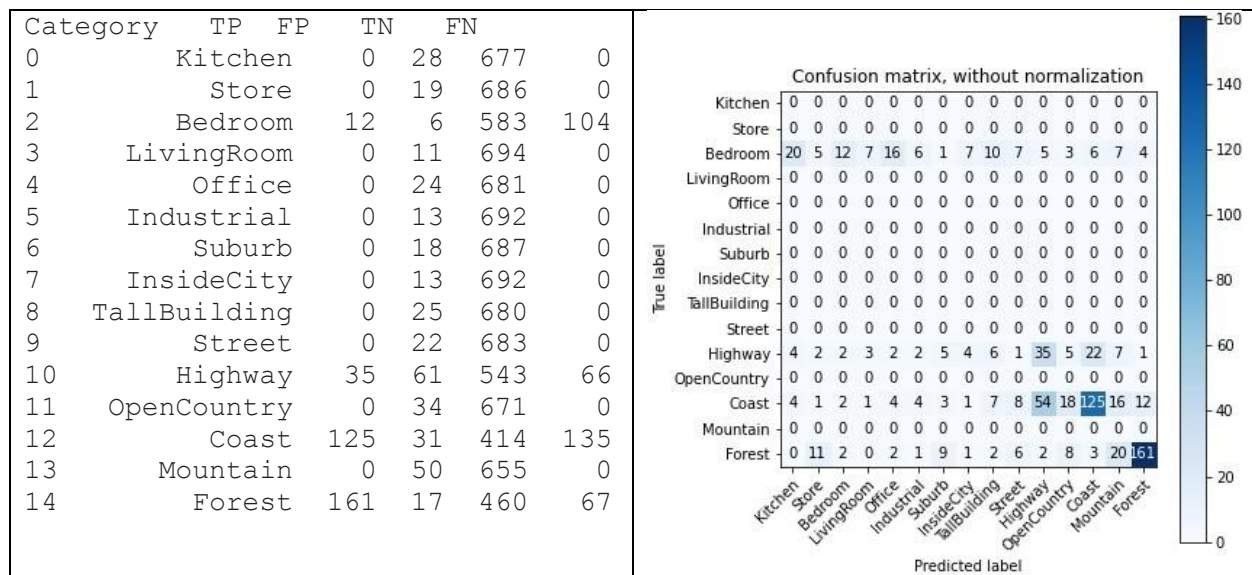
Category	TP	FP	TN	FN	
0	Kitchen	0	20	685	0
1	Store	0	16	689	0
2	Bedroom	39	2	587	77
3	LivingRoom	0	5	700	0
4	Office	0	18	687	0
5	Industrial	0	10	695	0
6	Suburb	0	34	671	0
7	InsideCity	0	11	694	0
8	TallBuilding	0	22	683	0
9	Street	0	19	686	0
10	Highway	37	52	552	64
11	OpenCountry	0	25	680	0
12	Coast	119	33	412	141
13	Mountain	0	42	663	0
14	Forest	185	16	461	43

Confusion matrix, without normalization

Kitchen	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Store	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bedroom	19	3	39	2	18	1	8	5	3	4	1	0	8	5	0				
LivingRoom	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Office	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Industrial	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Suburb	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
InsideCity	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TallBuilding	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Street	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Highway	0	0	1	2	0	2	3	2	9	6	37	12	23	3	1				
OpenCountry	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Coast	1	2	1	0	0	6	14	4	10	7	51	11	119	19	15				
Mountain	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Forest	0	11	0	1	0	1	9	0	0	2	0	2	2	15	185				

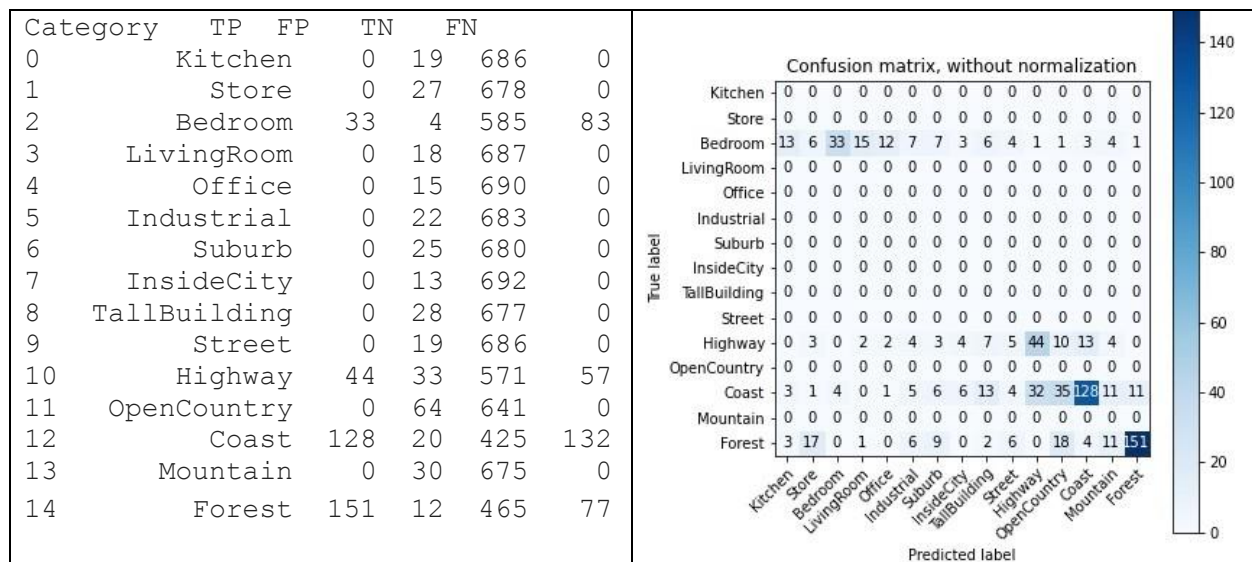
SVM with vocab size = 100 and descriptors max=200 random_state=2, tol=1e-05

Accuracy = 47%



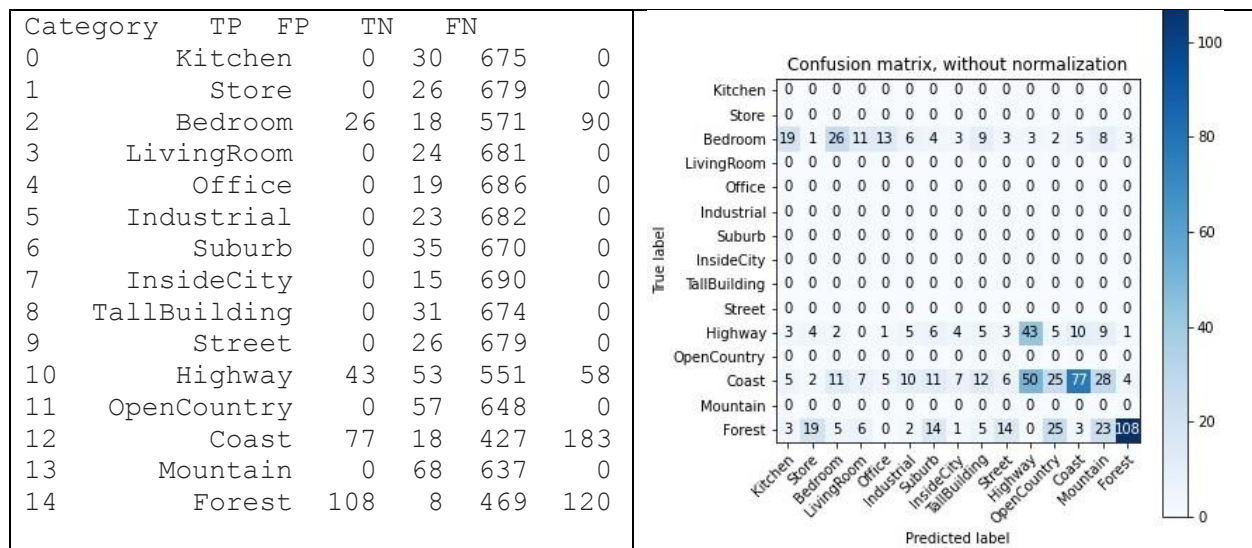
SVM with vocab size = 200, descriptors max=50,250

Accuracy= 50%



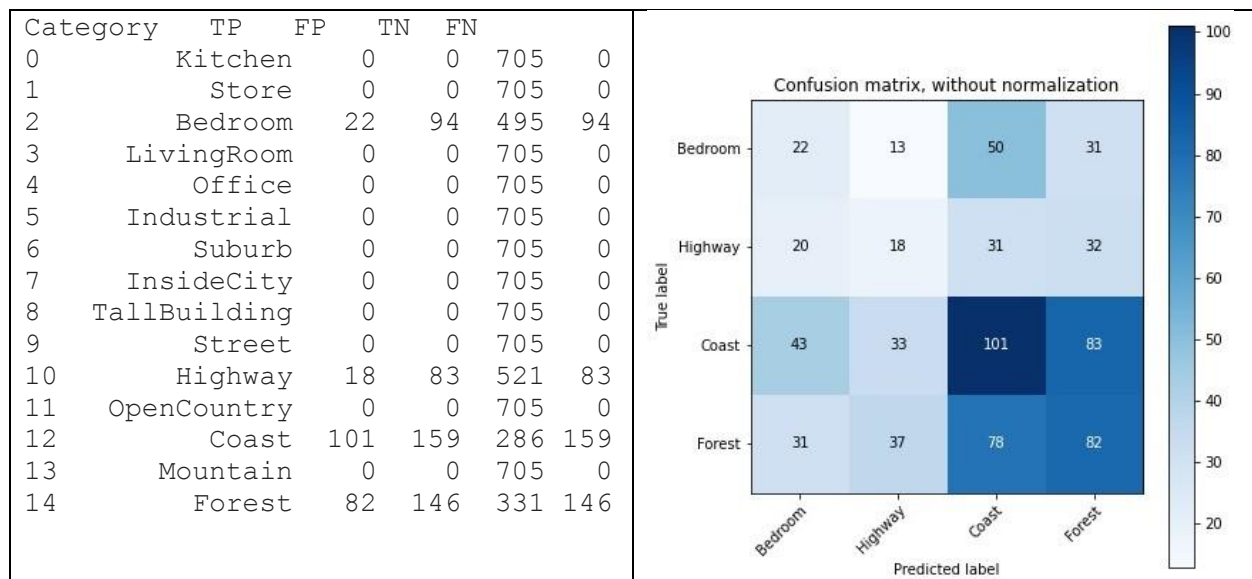
SVM with vocab size = 400, descriptors max=200

Accuracy = 36%



Placeholder.....starter code classifier

The starter code when run as it is i.e without any vocabulary or built up image features and proper classifiers, does permutations on the test labels and then randomly chose the categories from test labels as prediction. The accuracy is 31% percent.



Reading Part

Question 2: How is the proposed approach different and beneficial.

Answer: Simple bag of words is an orderless representation of the images i.e no special information is captured whereas the proposed approach in this paper considers the spacial information of a key point in an image. It divides the image into sub rejoin and then constructs histograms of local features inside each sub region. This way it matches two images based upon geometric correspondences.

It is computationally very efficient and far more accurate, 80%, than the simple bag of words model,50-60% and has shown state of the art performance on Caltech-101 database.

Question 4: Kind of features used.

Answer: Two kind of features are used for experiments. First type are named 'weak features' that are points whose gradient magnitude in a given direction is greater than some minimum threshold. They also call it edge points.

Another one are named as 'strong features', which are Sift descriptors of 16x16 pixel patches computed over a grid with 8 pixels spacing. Dense features work better for scene classification that is why dense regular grid was used instead of interest points.
