CS423/523 Computer Vision Fall 2015 [Project 3 and 4 Combined]

ERDEM YEREBASMAZ
S000969
DEPARTMENT OF COMPUTER SCIENCE

NEKRUZJON MAXUDOV
S001015
DEPARTMENT OF COMPUTER SCIENCE

Introduction

In this project, we were tasked to develop an OpenCV program using C++ that can recognize scene category of a give images. We have implemented 2 types of image representation techniques:

- Tiny image representation
- Bag of SIFT representation

After extracting features from images we have used *k*-NN and SVM classification techniques to classify the scenes of any given image.

Naming among training and test data folders were not consistent, therefore we edited them to make data reading easier.

Tiny Images with *k*-NN classifier

We have scaled the scene images to several low resolution sizes of 8x8, 16x16 and 32x32. After that, we have applied zero mean and normalization on these resized images. The resized 2d image array is then reshaped into a single dimensional array that has NxN elements.

The effect of *k* value and resolution size on accuracy can be seen in the tables below:

k =	1	3	5	10	15
Accuracy	**23.08 %**	21.21 %	21.88 %	22.04%	22.51 %

Table 1: Results for 64 dimensional tiny image feature vectors.

k =	1	3	5	10	15
Accuracy	**23.52 %**	21.54 %	21.84 %	22.14 %	22.55 %

Table 2: Results for 256 dimensional tiny image feature vectors.

k =	1	3	5	10	15
Accuracy	**22.61 %**	20.74 %	21.51 %	21.61 **%**	22.48 %

Table 3: Results for 1024 dimensional tiny image feature vectors.

Since the data cloud is not seperable good enough increasing *k* value does not improve accuracy rate. To test our theory we run simulation for 5 *k*-values and we have observed that accuracy fluctuated between **20%** and **24%**. Without normalization there was a loss of accuracy between 5% and 8%. We got the best results for ***k*=1** and tiny image of size **16x16**, normalized.

Bag of SIFT with *k*-NN classifier

Bag of SIFT

We have implemented several methods to extract vocabulary. We'll explain these methods and elaborate on the results and performance of each method.

Method 1

- Extract SIFT features using xfeatures2d library for all training set
- Use *k* means to cluster and compute vocabulary
- Using obtained vocabulary in previous stage compute histograms for training and test set
- Use *k*-NN & SVM classification techniques to classify images

Method 2

- Extract fixed amount of SIFT features from 4 different quadrants *(upper left, upper right, bottom left, bottom right)* of each image
- Use *k* means to cluster and compute vocabulary
- Using obtained vocabulary in previous stage compute histograms for training and test set *(however, compute SIFT features for whole image without seperating into quadrants)*
- Use *k*-NN & SVM classification techniques to classify images

Method 3

- Fit the regular grid onto image
- For each grid center, compute KeyPoints
- Using those KeyPoints describe each quadrant with SIFT descriptor
- Use *k* means to cluster and compute vocabulary
- Before computing histograms repeat first three steps with a grid twice the density
- Using obtained vocabulary in previous stage compute histograms for training and test set
- Use *k*-NN & SVM classification techniques to classify images

Results obtained using Bag of SIFT(Method 3) with *k*-nn classifier

k =	1	5	10	20	30
Accuracy	47.24 %	49.25 %	**51.26 %**	50.72 %	50.59 %

Table 1: Results for vocabulary size of 50.

k =	1	5	10	20	30
Accuracy	50.05 %	52.13 %	**53.80 %**	53.30 %	53.20 %

Table 2: Results for vocabulary size of 100.

k =	1	5	10	20	30
Accuracy	49.88 %	54.07 %	**55.18 %**	53.84 %	53.97 %

Table 3: Results for vocabulary size of 200.

k =	1	5	10	20	30
Accuracy	51.46 %	53.84 %	55.91 %	56.85 %	**57.22 %**

Table 4: Results for vocabulary size of 400.

As dimentionality increases, both information and noise increase. For high dimensional vocabularies using PCA for dimensionality reduction might increase the accuracy of *k*-NN since we can reduce the noise effectively.

Bag of SIFT with 1-vs-all linear SVM classifier

Before we implemented 1-vs-all SVM classification we have tested the data using all-vs-all SVM classifier and got **59 %** accuracy after parameter tuning. However, in the assignment we were asked to do 1-vs-all classification. While doing 1-vs-all classification using the same vocabulary, we have obtained accuracy around 40% then we changed our vocabulary creation method and used Method 2. Results using Method 2 can be seen in Results of Previous Methodologies part. We couldn't reach the expected performance with Method 2 and used Method 3.

For classification, we use classification scores returned by the SVM predictor. We store these scores in a result matrix and select the maximum score out of the results we obtained from the 15 1-vs-all binary classifiers.

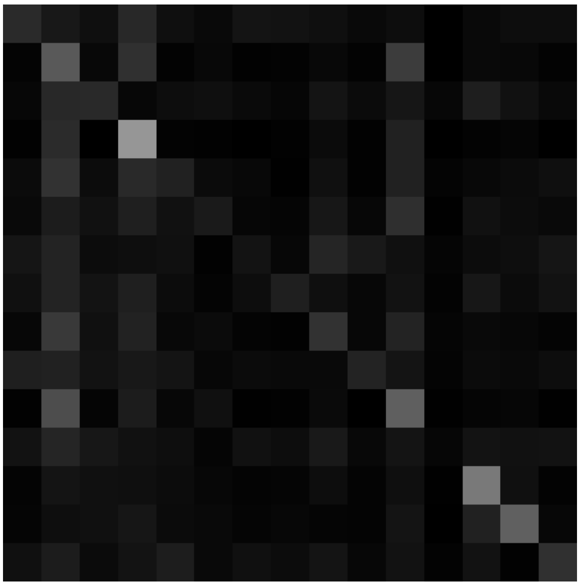
Results obtained using Bag of SIFT(Method 3) with 1-vs-all linear SVM classifier

Vocabulary Size	Accuracy
50	56.31 %-
100	59.36 %
200	60.50%
400	**61.68 %**

Confusion Matrices, Positive and Negative Instances

Tiny Images with k-NN classifier

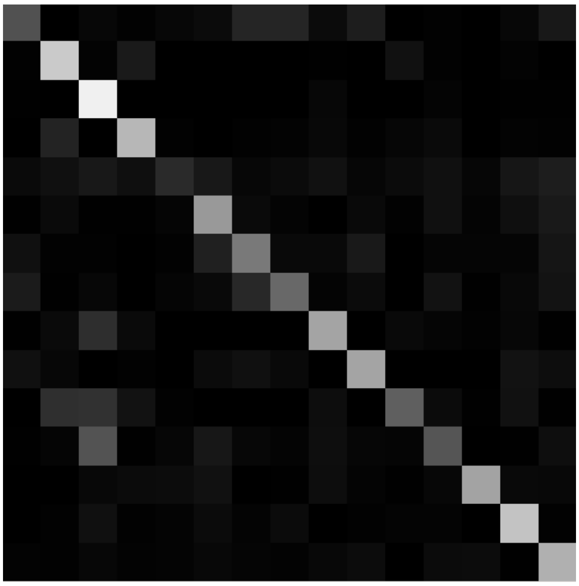
Confusion Matrix



Tiny Images with *k*-NN classifier. Accuracy of 23.52%
k: 1

Bag of SIFT with k-NN classifier

Confusion Matrix



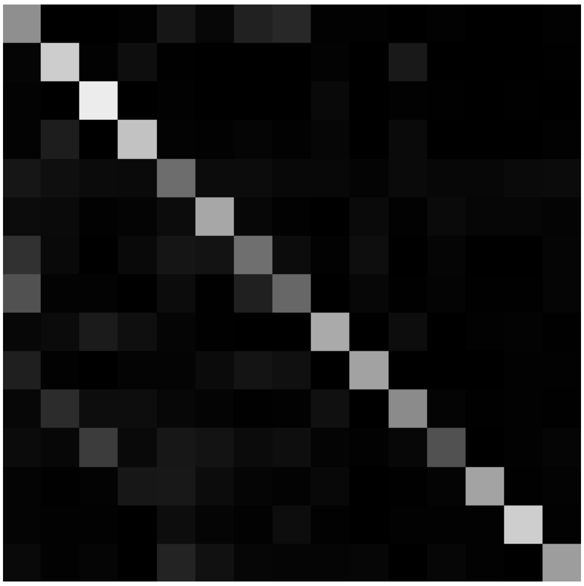
Bag of SIFT with k-NN classifier. Accuracy of 57.22%
Vocabulary Size: 400
k: 30

Positive and Negative Instances

	Bedroom	Coast	Forest	Highway	Industrial	Inside City	Kitchen	Living Room	Mountain	Office	Open Country	Store	Street	Suburb	Tall Building
True Positives	32%	79%	94%	72%	17%	60%	47%	41%	64%	64%	37%	33%	64%	77%	69%
False Negatives	68%	21%	6%	28%	83%	40%	53%	59%	36%	36%	63%	67%	36%	23%	31%
False Positives	31%	54%	99%	36%	19%	58%	50%	36%	41%	41%	22%	43%	12%	46%	62%

Bag of SIFT with 1-vs-all linear SVM classifier

Confusion Matrix



Bag of SIFT with 1-vs-all linear SVM classifier. Accuracy of 61.68%
Vocabulary Size: 400

Positive and Negative Instances

	Bedroom	Coast	Forest	Highway	Industrial	Inside City	Kitchen	Living Room	Mountain	Office	Open Country	Store	Street	Suburb	Tall Building
True Positives	56%	80%	93%	76%	43%	65%	44%	40%	67%	63%	55%	32%	64%	81%	62%
False Negatives	44%	20%	7%	24%	57%	35%	56%	60%	33%	37%	45%	68%	36%	19%	38%
False Positives	97%	53%	50%	40%	72%	43%	53%	46%	26%	18%	29%	18%	8%	12%	17%

Results of Previous Methodologies

Bag of SIFT and k-NN using Method 2

k =	**1**	**5**	**10**	**20**	**30**
Accuracy	39.10 %	42.11 %	44.39 %	44.76 %	**45.09** %

Table 1: Results for vocabulary size of 50.

k =	**1**	**5**	**10**	**20**	**30**
Accuracy	41.54 %	44.42 %	46.50 %	46.73%	**46.90** %

Table 2: Results for vocabulary size of 200.

k =	**1**	**5**	**10**	**20**	**30**
Accuracy	38.59 %	43.75 %	44.15 %	**44.46 %**	43.48 %

Table 3: Results for vocabulary size of 300.

Bag of SIFT and linear SVM using Method 2

Vocabulary Size	**Accuracy**
50	54.49 %
200	**54.57 %**
300*	53.47 %
400	53.50 %

*: *We have worked with vocabulary sizes of 50,200,300,400 unlike 50,100,200,400 in Method 3.*

Bag of SIFT and linear SVM using Method 3 with random sampling**

Vocabulary Size	**Accuracy**
50	~56.31 %
100	~57.96 %
200	~58.53 %
400	**~61.37 %**

**: *Results are the average of 10 runs, although it was not recorded standart deviation varied between 0.40% and 0.60%*