CENG 483

Introduction to Computer Vision

Fall 2021-2022

Take Home Exam 2

Object Recognition

Student ID: 2098705

Please fill in the sections below only with the requested information. If you have additional things to mention, you can use the last section. Please note that all of the results in this report should be given for the **validation set**. Also, when you are expected to comment on the effect of a parameter, please make sure to fix other parameters.

1 Local Features (25 pts)

- Explain SIFT and Dense-SIFT in your own words. What is the main difference?

  In SIFT implementation, we can obtain key features of an object which are at some specific locations on the object such as corners,edges etc but in Dense-SIFT, this will be more detailed extraction rather than SIFT.Dense-SIFT can extract descriptors from every region not just specific location like SIFT.This searching and identifying size in Dense-SIFT can be arranged with step size values.

- Put your quantitative results (classification accuracy) regarding 5 values of SIFT and 3 values of Dense-SIFT parameters here. In SIFT change each parameter once while keeping others same and in Dense-SIFT change size of feature extraction region. Discuss the effect of these parameters by using 128 clusters in k-means and 8 nearest neighbors for classification.

  With default configurations of SIFT:
  nfeatures = 0, nOctaveLayers = 3,contrastThreshold = 0.04,
  edgeThreshold = 10,sigma = 1.6
  Accuracy: %17.1

  With changing parameter nfeatures values :
  nfeatures = 1 - %18.42
  nfeatures = 2 - %17.96
  nfeatures = 3 - %16.49
  nfeatures = 4 - %18.76
  In this parameter, SIFT retains the number of best features depends on nfeatures values.When I increased nfeatures values until 4, I obtained more accurate results but we can say that after some value of nfeatures, it will detect noisy points as key points and it leads to classify images falsely.

  With changing parameter nOctaveLayers values :

nOctaveLayers = 1 - %16.89
nOctaveLayers = 2 - %17.02
nOctaveLayers = 4 - %17.36
nOctaveLayers = 5 - %18.02
When increasing nOctaveLayers, key points detected in the image increases.When comparing the default value which is 3, I obtained accuracy values which are directly proportional with nOctaveLayers values which means it needs to detect more key points to get higher accuracy values.

With changing parameter contrastThreshold values :
contrastThreshold = 0.01 - %15.82
contrastThreshold = 0.02 - %18.82
contrastThreshold = 0.03 - %18.22
contrastThreshold = 0.05 - %17.36

contrastThreshold value filters out weak features in the image and when its value is increased lower features are produced.When I decreased the default value which is 0.04, it produced more accurate results that means image classifier needs more features for successful prediction.

With changing parameter edgeThreshold values :
edgeThreshold = 8 - %16.76
edgeThreshold = 9 - %16.56
edgeThreshold = 11 - %18.76
edgeThreshold = 12 - %17.02

edgeThreshold value is like opposite of contrastThreshold value.When it is increasing less filtering is applied.Therefore, after I increased the default value which is 10, accuracy increased when edgeThreshold = 11, as expected.

With changing parameter sigma values :
sigma = 1.4 - %16.29
sigma = 1.5 - %17.49
sigma = 1.7 - %18.42
sigma = 1.8 - %16.36

Dense-SIFT Accuracy Values:
step size = 3 - %9.41
step size = 5 - %10.35
step size = 7 - %9.61

Dense-SIFT algorithm chooses some grids to find key points and features according to step size value.When we increase step size,the grid portion decreases and detected key points and features will decrease as well.In this experiment, I can say that step size = 5 is a critical value and increasing or decreasing this value makes algorithm's classifying capability lower.We can conclude from this tuning, the grid size for detection key points and features should not be much lower and much higher because when higher step sizes are chosen, it will detect unnecessary and irrelevant key points and features and making true predictions for labels will be more hard.Again, when lower step sizes are chosen, it will detect less key points and features than it needs and it will not be sufficient to classify successfully.

2 Bag of Features (45 pts)

- How did you implement BoF? Briefly explain.

  Firstly, we need to extract local descriptors in the data set using SIFT or Dense-SIFT algorithm.I extracted all local descriptors in the data set and gathered them in a numpy array.After that, I applied Kmeans clustering algorithm to this numpy array and centers of this algorithm's returned value are our visual words.This process until now is forming the dictionary.From this point, I created histograms one by one for all images in the training data set using visual words and gathered these histograms in a list.After this process, the validation datas should have been used.For all images in the validation data set, I created histograms again using visual words.For every histograms in validation set, I applied KNN algorithm with histogram list formed one step before to predict labels of validation images.Most dominant labels of neighbours of validation image's histogram with smallest distances determined by k value are the label prediction of our validation image.

- Give pseudo-code for obtaining the dictionary.

  imageSet = x_1,x_2,.....,x_n
  descriptorList = [ ]
  for i = 1 to n

  descpriptor = SIFT(x)
  descriptorList.pushBack(descriptor)

  kmean = KMean(k,descriptorList)
  dictionary = kmean.getCenters()

- Give pseudo-code for obtaining BoF representation of an image once the dictionary is formed.

  im = getImage()
  descriptor = SIFT(image)
  BoFRep = list.zeros(dictionary.size())
  for i = 1 to dictionary.size()

  for k = 1 to descriptor.size()
  if(descriptor[k] == dictionary[i])
  BoFRep[k] += 1

  BoFRep = BoFRep.normalize()

- Put your quantitative results (classification accuracy) regarding 3 different parameter configurations for the BoF pipeline here. Discuss possible reasons for each one's relatively better/worse accuracy. You are suggested to keep $k \leq 1024$ in k-means to keep experiment durations managable. You need to use the best feature extractor you obtained in the previous part together with the same classifier.

  Because I obtained better accuracy results of SIFT with parameter edgeThreshold = 11, I continued with this configuration.
  With different KMeans k values:

  k = 256 - %15.95
  k = 512 - %14.96

k = 64 - %18.95
k = 56 - %20.56

Since we determine visual words by tuning k values of Kmeans, when we increase the number of centers too much, algorithm chooses unnecessary center points that may cause to find misclassified labels due to noisy centers.This situation makes algorithm to find meaningless and incoherent "likeness" between two images in different classes.Therefore, if we tune the k value higher from its optimum value, the accuracy decreases.With value of k = 56, we get better accuracy so that this shows that k = 56 is more suitable value to make a cluster than k = 128 in the aim of obtaining better accuracies.We also can say that because the aim of Kmeans algorithm to find minimum objective function value, with k = 56 we are getting lower objective function value.

3 Classification (30 pts)

- Put your quantitative results regarding k-Nearest Neighbor Classifier for k values 16, 32 and 64 by using the best k-means representation and feature extractor. Discuss the effect of these briefly.

  k = 16 - %18.22
  k = 32 - %19.22
  k = 64 - %20.02
  Increasing k values makes algorithm to find neighbours of given image with same classes easier until k = 64 but it does not mean accuracy will increase with k value's increasing because when we exceed again optimum k value accuracy starts to decrease because predicted label tends to shift dominant class in the found neighbours and if we decrease k value from its optimum, then this time again accuracy will decrease because algorithm will become more sensitive to noisy datas.

- What is the accuracy values, and how do you evaluate it? Briefly explain.

  Simply, I tested the trained algorithm with predicted labels and compare them with actual labels of data in the validation set.After finishing of testing process, number of obtained correctly classified objects divided by number of all classified objects gives us the accuracy value.

- Give confusion matrices for classification results of these combinations.

  In these confusion matrices, I write accuracy values in percentage.

Figure 1: Confusion matrix of k = 16

Predicted

Actual	Apple	Aquarium Fish	Beetle	Camel	Crab	Cup	Elephant	Flatfish	Lion	Mushroom	Orange	Pear	Road	Skyscraper	Woman
Apple	25.0	18.0	7.0	10.0	9.0	1.0	3.0	5.0	3.0	7.0	5.0	1.0	2.0	0.0	4.0
Aquarium Fish	2.0	28.0	2.0	7.0	5.0	2.0	9.0	10.0	7.0	14.0	2.0	4.0	3.0	0.0	5.0
Beetle	8.0	4.0	43.0	9.0	3.0	5.0	10.0	6.0	6.0	1.0	2.0	2.0	0.0	0.0	1.0
Camel	2.0	4.0	9.0	16.0	8.0	1.0	21.0	8.0	10.0	11.0	2.0	4.0	1.0	0.0	3.0
Crab	5.0	7.0	9.0	15.0	18.0	1.0	9.0	1.0	13.0	10.0	2.0	3.0	2.0	1.0	4.0
Cup	3.0	15.0	9.0	14.0	7.0	5.0	9.0	10.0	10.0	9.0	3.0	0.0	3.0	3.0	0.0
Elephant	2.0	5.0	12.0	19.0	17.0	0.0	13.0	9.0	12.0	4.0	1.0	4.0	8.0	1.0	1.0
Flatfish	3.0	12.0	6.0	12.0	10.0	2.0	7.0	13.0	14.0	7.0	4.0	3.0	2.0	1.0	4.0
Lion	1.0	8.0	5.0	14.0	5.0	0.0	7.0	10.0	26.0	11.0	1.0	5.0	1.0	0.0	6.0
Mushroom	1.0	19.0	4.0	9.0	7.0	0.0	8.0	6.0	10.0	28.0	4.0	2.0	1.0	0.0	1.0
Orange	13.0	7.0	7.0	10.0	7.0	1.0	9.0	7.0	11.0	10.0	6.0	4.0	2.0	2.0	4.0
Pear	8.0	13.0	14.0	7.0	4.0	7.0	2.0	12.0	11.0	11.0	4.0	0.0	3.0	0.0	4.0
Road	2.0	9.0	6.0	7.0	8.0	0.0	5.0	8.0	8.0	4.0	3.0	2.0	30.0	1.0	7.0
Skyscraper	4.08	10.2	5.1	4.08	4.08	5.1	5.1	10.2	12.24	9.18	2.04	5.1	13.26	6.12	4.08
Woman	5.0	12.0	4.0	11.0	9.0	1.0	9.0	11.0	17.0	11.0	2.0	2.0	2.0	0.0	4.0

Figure 2: Confusion matrix of k = 32

Predicted

	Apple	Aquarium Fish	Beetle	Camel	Crab	Cup	Elephant	Flatfish	Lion	Mushroom	Orange	Pear	Road	Skyscraper	Woman
Apple	20.0	13.0	8.0	9.0	10.0	2.0	11.0	4.0	9.0	4.0	6.0	1.0	2.0	0.0	1.0
Aquarium Fish	0.0	37.0	1.0	13.0	1.0	1.0	8.0	4.0	11.0	19.0	1.0	1.0	1.0	0.0	2.0
Beetle	4.0	1.0	46.0	17.0	4.0	2.0	9.0	6.0	5.0	3.0	0.0	1.0	1.0	1.0	0.0
Camel	0.0	3.0	8.0	24.0	4.0	0.0	19.0	11.0	11.0	9.0	1.0	4.0	4.0	2.0	0.0
Crab	4.0	9.0	7.0	13.0	11.0	2.0	12.0	7.0	19.0	11.0	1.0	2.0	0.0	2.0	0.0
Cup	4.0	14.0	14.0	11.0	5.0	5.0	5.0	7.0	8.0	10.0	1.0	1.0	5.0	3.0	7.0
Elephant	2.0	2.0	4.0	24.0	12.0	1.0	18.0	10.0	14.0	10.0	1.0	1.0	0.0	0.0	1.0
Flatfish	2.0	8.0	7.0	13.0	4.0	2.0	16.0	13.0	14.0	9.0	0.0	2.0	5.0	2.0	3.0
Lion	0.0	8.0	5.0	15.0	4.0	0.0	13.0	6.0	23.0	21.0	0.0	0.0	1.0	0.0	4.0
Mushroom	0.0	24.0	2.0	5.0	4.0	0.0	9.0	7.0	13.9	33.0	0.0	0.0	2.0	0.0	1.0
Orange	8.0	4.0	7.0	17.0	1.0	1.0	11.0	6.0	20.0	8.0	6.0	3.0	2.0	1.0	5.0
Pear	7.0	16.0	10.0	19.0	4.0	1.0	9.0	6.0	9.0	10.0	1.0	1.0	2.0	2.0	3.0
Road	1.0	6.0	6.0	8.0	4.0	2.0	7.0	5.0	6.0	5.0	1.0	0.0	37.0	7.0	5.0
Skyscraper	3.06	10.2	5.1	7.14	7.14	1.02	9.18	6.12	13.26	8.16	3.06	1.02	13.26	9.18	3.06
Woman	2.0	11.0	3.0	16.0	3.0	0.0	17.0	13.0	17.0	9.0	2.0	2.0	4.0	0.0	1.0

Figure 3: Confusion matrix of k = 64

Predicted

Actual	Apple	Aquarium Fish	Beetle	Camel	Crab	Cup	Elephant	Flatfish	Lion	Mushroom	Orange	Pear	Road	Skyscraper	Woman
Apple	16.0	15.0	8.0	7.0	6.0	0.0	12.0	2.0	8.0	15.0	2.0	1.0	4.0	0.0	4.0
Aquarium Fish	2.0	35.0	1.0	3.0	1.0	0.0	5.0	5.0	22.0	16.0	1.0	1.0	1.0	0.0	7.0
Beetle	6.0	1.0	43.0	19.0	2.0	1.0	4.0	3.0	8.0	4.0	0.0	1.0	2.0	1.0	5.0
Camel	0.0	5.0	6.0	17.0	4.0	1.0	17.0	13.0	18.0	8.0	1.0	0.0	5.0	0.0	5.0
Crab	1.0	8.0	9.0	15.0	8.0	1.0	10.0	8.0	25.0	11.0	0.0	0.0	2.0	1.0	1.0
Cup	1.0	14.0	7.0	10.0	11.0	2.0	4.0	11.0	11.0	15.0	2.0	2.0	5.0	2.0	3.0
Elephant	0.0	2.0	9.0	22.0	7.0	1.0	21.0	8.0	20.0	5.0	0.0	0.0	1.0	0.0	4.0
Flatfish	1.0	8.0	5.0	15.0	8.0	0.0	13.0	15.0	15.0	6.0	0.0	2.0	6.0	2.0	4.0
Lion	0.0	3.0	6.0	10.0	3.0	0.0	11.0	6.0	29.0	23.0	1.0	3.0	0.0	0.0	5.0
Mushroom	0.0	21.0	0.0	5.0	3.0	1.0	6.0	5.0	20.0	31.0	1.0	1.0	2.0	0.0	4.0
Orange	8.0	6.0	5.0	15.0	2.0	0.0	9.0	5.0	24.0	11.0	5.0	4.0	2.0	1.0	3.0
Pear	5.0	15.0	8.0	12.0	7.0	1.0	7.0	3.0	16.0	10.0	1.0	3.0	4.0	0.0	8.0
Road	1.0	3.0	3.0	11.0	0.0	1.0	7.0	6.0	11.0	7.0	1.0	2.0	32.0	5.0	10.0
Skyscraper	0.0	13.26	1.02	9.18	5.1	3.06	10.2	9.18	12.24	9.18	1.02	2.04	14.28	6.12	4.08
Woman	0.0	10.0	1.0	7.0	4.0	0.0	13.0	11.0	24.0	13.0	2.0	2.0	4.0	1.0	8.0

4 Additional Comments and References

(if there any)