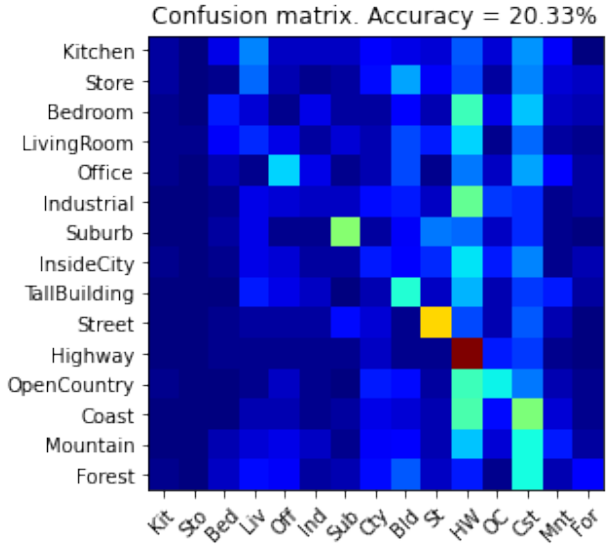
CS 4476
PS 5

Jinghong Peng
jpeng78@gatech.edu
903568613

Part 1: Tiny Image Representation and Nearest-Neighbor Classification

Part 1.3.a: Your confusion matrix, together with the accuracy for Part 1 with the standard parameter set (image_size = 16, k = 3)

<Plot here>



Part 1.3.b: Experiments: change image size and k individually using the following values, and report the accuracy (when tuning one parameter, keep the other as the standard (16 x 16, 3)):

ie. when you're tuning image size, keep k at 3, when changing k, keep image size as 16x16

image size:

8 x 8: 16.67%

16 x 16: 20.33%

32 x 32: 20.93%

k:

1: 20.87%

3: 20.33%

5: 22.93%

10: 22.07%

15: 22.53%

Part 1.3.c: When tuning the parameters (image size and k), what did you observe about the *processing time and accuracy*? What do you think led to this observation?

As I increase the size of the image and increase K, the processing time increases. So I think larger size and larger K increase processing time. Larger image size will result in higher accuracy. Accuracy increase as K increases, but after crossing the optimal K value, the accuracy start to decrease.

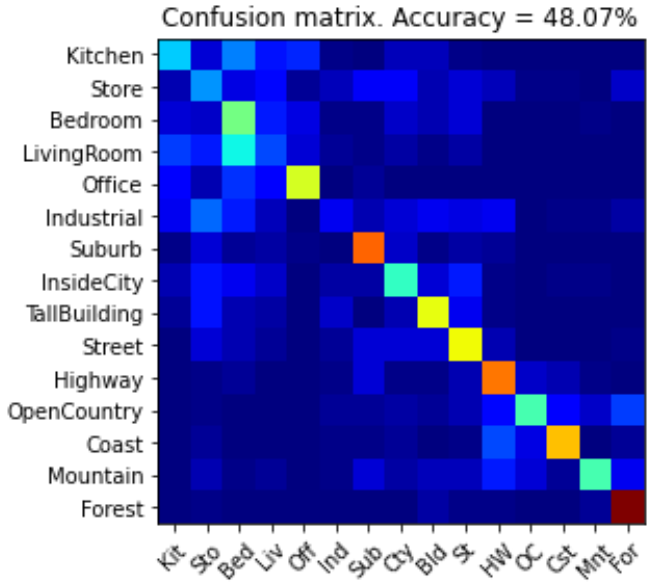
Part 2: Bag-of-words with SIFT Features

Part 2.3: Reflection on Tiny Image Representation vs. Bag of Words with SIFT features:

Why do you think that the tiny image representation gives a much worse accuracy than bag of words? Additionally why do you think Bag of Words is better in this case?

Tiny image resize the image to lower resolution, thus lose information about the image. Bag of words keep important features of the image. Bag of words utilize most of the data for clustering.

Part 2.4.a: Your confusion matrix, together with the accuracy for Part 2 with the standard parameter set (vocab_size = 50, k = 3, max_iter = 10, stride(build_vocab) = 20, stride(get_bags_of_sift) = 5



Part 2.4.a: Experiments: change vocab_size and k individually using the following values, and report the accuracy (when tuning one parameter, keep the other as the standard (50, 3)):

ie. when you're tuning vocab_size, keep k at 3, when changing k, keep vocab_size as 50. (Other params max_iter = 10, stride(build_vocab) = 20, stride(get_bags_of_sift) = 5)

vocab size:

50: 48.07%

100:48.07%

200:48.04%

k:

1: 48.07%

3: 48.07%

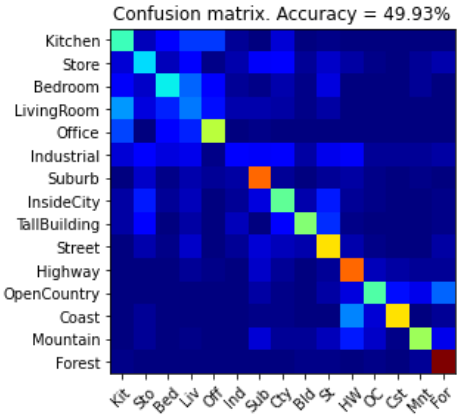
5:49.93%

10:48.73%

15:48.00%

Part 2.4.a: Paste the confusion matrix for your best result with the previous experimentation in this slide.

<Plot here>



vocab_size:

k:

max_iter: 10

stride(build_vocab): 20

stride(get_bags_of_sift): 5

Part 2.4.b: Reflection: when experimenting with the value k in kNN, what did you observe? Compare the performance difference with the k value experiment in Part 1.3, what can you tell from this?

Increase K increase accuracy slightly, and after the optimal K accuracy decrease slightly. It is similar to the effect of k in Part 1.3

Part 3: Extra Credit

EXTRA CREDIT

Part 3.1: Post best confusion matrix, together with the accuracy out of all the parameters you tested. Report the parameter settings used to obtain this result.

<Plot here>

Parameter settings:

max_iter:

stride(build_vocab):

stride(get_bags_of_sift):

vocab_size:

k (kNN):

EXTRA CREDIT

Part 3.2: Post confusion matrix along with the distance metric that you used for achieving a better accuracy on standard parameters. Why do you think it performs better?

<Plot here>

Distance metric and why it works better:

EXTRA CREDIT

Part 3.3: Post confusion matrix along with your explanation of your SVM model and detail any other changes your made to reach an accuracy of 65% or greater.

<Plot here>

Description of your model: