HACETTEPE UNIVERSITY

Department of Computer Engineering
SPRING 2019

BBM 418 Computer Vision Laboratory Problem Set-2

Supervisors:	Dr. Nazlı Ikizler Cinbis
	TA Ozge Yalcinkaya
Author:	Ege Berke Balseven
Number:	21590776
Subject:	Image Classification with CNN
Due Date:	06.05.2019

Contents

1 Introduction

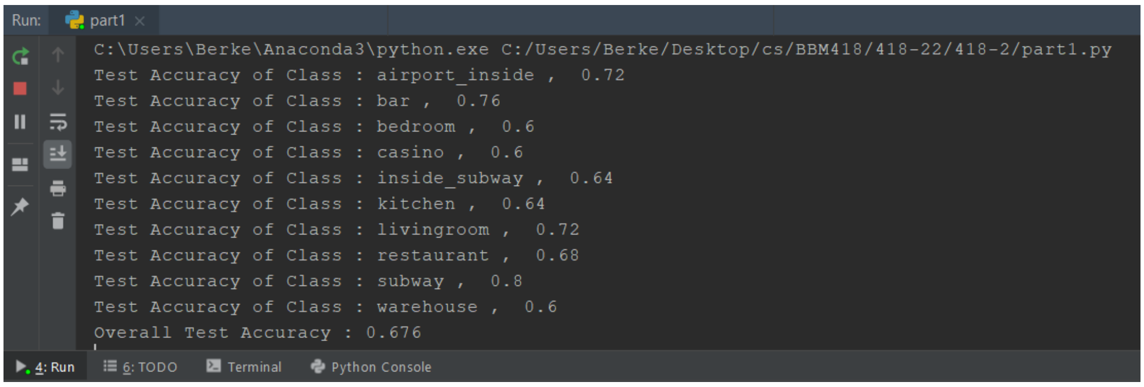
In this experiment, i got familiar use of the Convolutional Neural Network (CNN) and Fine Tuning on the Pre-Trained Model. Using pre-trained models better than do it from scratch while using CNN. In part1 i did not do fine tuning, directly used the pre-trained model and used SVM classification. In part 2 i did fine tuning and saved the model. Then last part, i loaded the model and i did things that in part1.

2 Implementation Details

I've had too many problems. Some of these are: svm classification, getting fc layers, loading saved model, torch gpu and ram runtime errors, cuda error, training validation set, top5 accuracy, plotting loss and accuracy table, confusion matrix etc. I spent a lot of hours solving all of these problems.
I used pre-trained VGG16 model from torch library trained on ImageNet. By replacing the fully-connected layers of these models with mine dataset classes, i tried to re-train this part of the model. I compared the Hyper-Parameters with different epoch numbers, different batch sizes or by changing the learning rate. ButIn part1 i am not train any weights, directly used the pre-trained VGG-16 model then i did multi-class linear SVM for classiffication and get class based/total accuracies. In part 2 i freeze the layers except FC layers for classification and i did fine tuning. Then in part 3, i loaded the fine tuned model of part2 and i did same things in part1 and got better accuracy then part1.

3 Experimental Results

3.1 PART 1: Image Representation



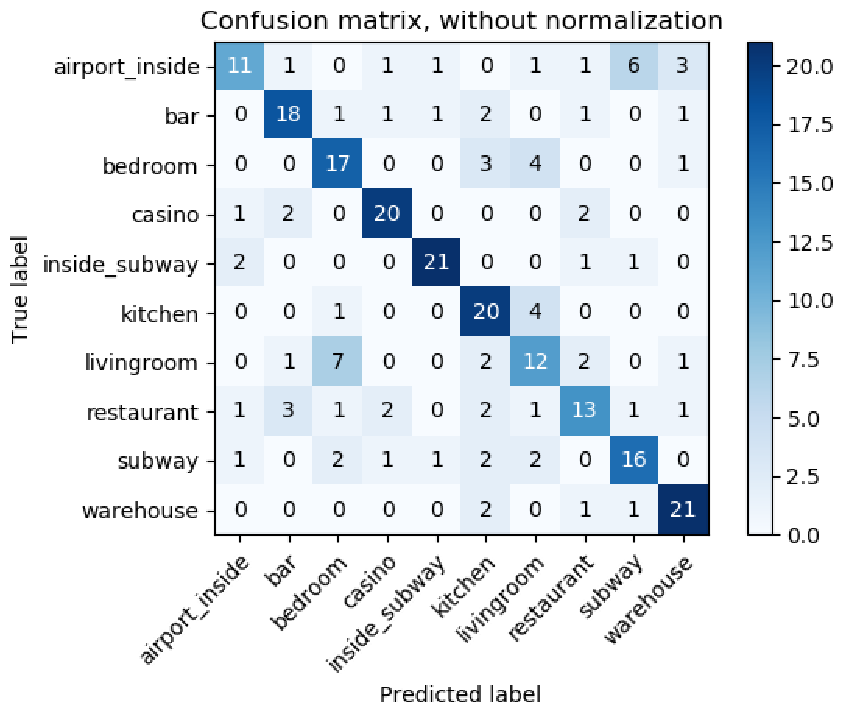
```
C:\Users\Berke\Anaconda3\python.exe C:/Users/Berke/Desktop/cs/BBM418/418-22/418-2/part1.py
Test Accuracy of Class : airport_inside ,  0.72
Test Accuracy of Class : bar ,  0.76
Test Accuracy of Class : bedroom ,  0.6
Test Accuracy of Class : casino ,  0.6
Test Accuracy of Class : inside_subway ,  0.64
Test Accuracy of Class : kitchen ,  0.64
Test Accuracy of Class : livingroom ,  0.72
Test Accuracy of Class : restaurant ,  0.68
Test Accuracy of Class : subway ,  0.8
Test Accuracy of Class : warehouse ,  0.6
Overall Test Accuracy : 0.676
```

overall test accuracy%67.6

Accuracy is lower than in other parts.

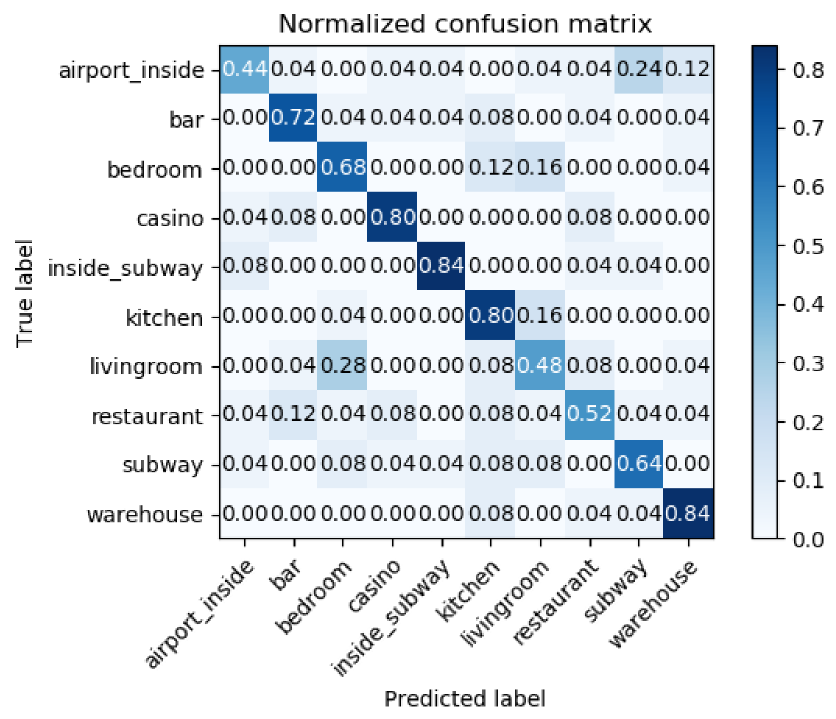
batch size = 25

num workers = 4



	airport_inside	bar	bedroom	casino	inside_subway	kitchen	livingroom	restaurant	subway	warehouse
airport_inside	11	1	0	1	1	0	1	1	6	3
bar	0	18	1	1	1	2	0	1	0	1
bedroom	0	0	17	0	0	3	4	0	0	1
casino	1	2	0	20	0	0	0	2	0	0
inside_subway	2	0	0	0	21	0	0	1	1	0
kitchen	0	0	1	0	0	20	4	0	0	0
livingroom	0	1	7	0	0	2	12	2	0	1
restaurant	1	3	1	2	0	2	1	13	1	1
subway	1	0	2	1	1	2	2	0	16	0
warehouse	0	0	0	0	0	2	0	1	1	21

confusion matrix



normalized confusion matrix

The confusion matrix shows the ways in which your classification model is confused when it makes predictions. It gives us insight not only into the errors being made by a classifier but more importantly the types of errors that are being made. In confusion matrix the diagonal elements represent the number of points for which the predicted label is equal to the true label, while off-diagonal elements are those that are mislabeled by the classifier. The higher the diagonal values of the confusion matrix the better, indicating many correct predictions.

By looking the confusion matrix, network bad at airport inside, livingroom, restaurant classes. Good at inside subway and warehouse classes. Restaurant confused with all classes except inside subway class. Airport inside confused with subway class a lot(%24). Also livingroom confused with bedroom a lot(%28) etc.

3.2 PART 2: Fine-tuning

Convolutional Neural Network (CNN):

Convolutional Neural Network The Neural Network is designed for image. Normally, if a normal Neural Network was used for images, it would be impossible to extract layers and features for a 300x300x3 image, for example. For this, methods like CNN Convolution, Pooling, and so on are used, which creates smaller data such as 3x3x3. This results in Fully-Connected Layers and is separated into classes by SoftMax. SoftMax analyzes the given values and finds out the closest percentages to the classes.

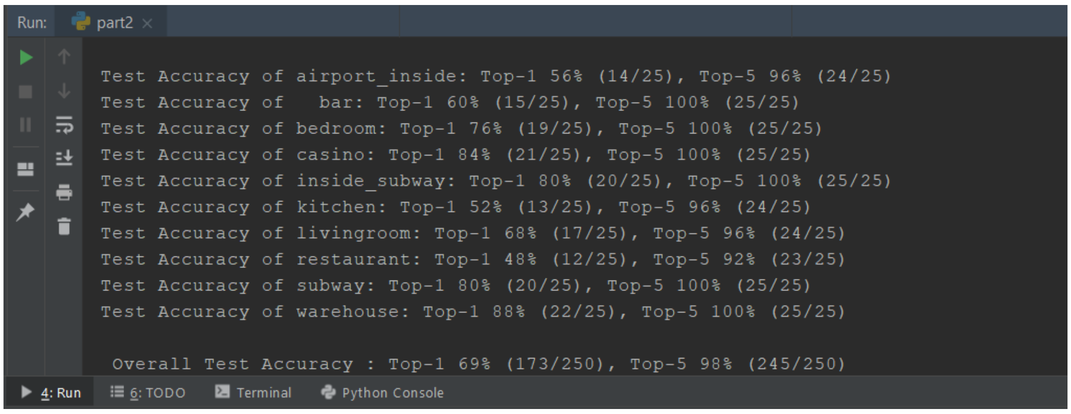
Fine-Tuning a Pre-Trained Model:

Deep neural networks like CNN usually have many parameters. Training CNN in a small dataset greatly affects CNN's generalization ability, often causing overfitting. For this reason, applications use train-ready pre-trained models in a larger data set. These models are models trained with 1.2 million images such as ImageNet. VGG, ResNet, AlexNet are examples. I can set my own dataset by separating the fully-connected layers of these ready-made models. So in ready-trained model my dataset will also be trained. This is called Fine Tuning. I used FC layers in part 2 because other layers use for extracting features, but FC layers use for classification.

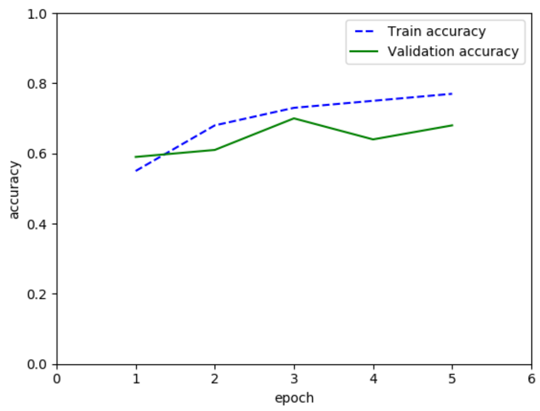
3.2.1 Comparison of Hyper-Parameters

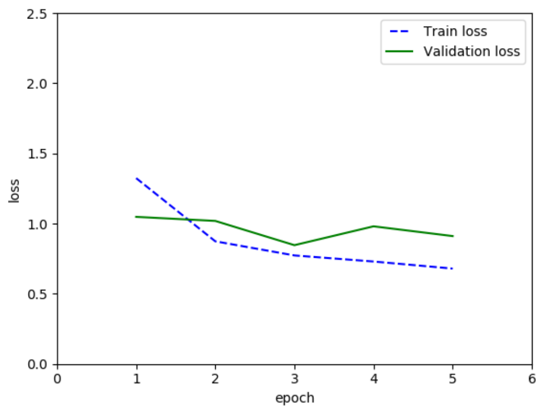
Number Of Epoch:

I have tried to observe the relation between Top1, Top5 Test Accuracy and Train, Validation Loss of epoch number using 25 batch size and 0.001 Learning rate with VGG16 model below. I can generally increase Top1 accuracy as the number of epoch increases. But the increase in Top5 Accuracy is very little observed. Because Top5 is always higher than Top1, the result may not always be the most probable, but it can be between the top 5 probabilities. So, Top5 always gets higher. When running with 5 Epoch, the Train Loss chart did not reach the desired look. With increasing epoch numbers Train loss is to fall to a certain extent and then remain little constant. Generally, increasing the number of Epochs can be better for the train. Because the more epoch the more neuron weights will update. So any forward and backpropagation will learn so much and the accuracy will rise.

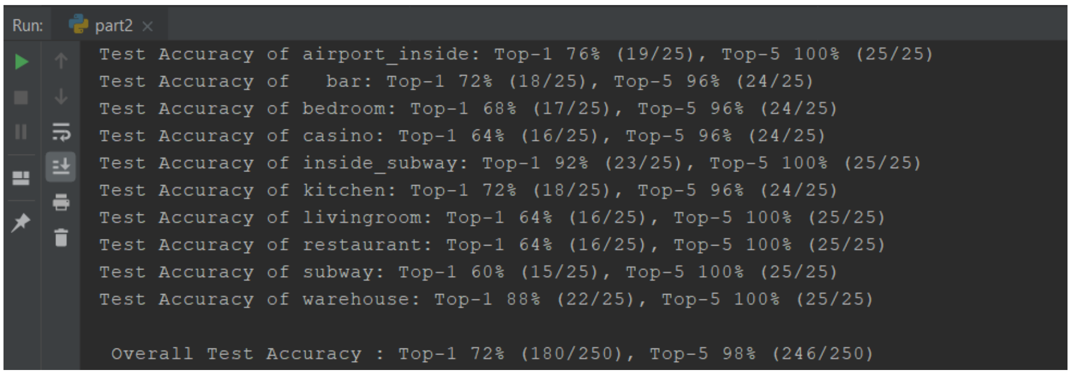


```
Test Accuracy of airport_inside: Top-1 56% (14/25), Top-5 96% (24/25)
Test Accuracy of   bar: Top-1 60% (15/25), Top-5 100% (25/25)
Test Accuracy of bedroom: Top-1 76% (19/25), Top-5 100% (25/25)
Test Accuracy of casino: Top-1 84% (21/25), Top-5 100% (25/25)
Test Accuracy of inside_subway: Top-1 80% (20/25), Top-5 100% (25/25)
Test Accuracy of kitchen: Top-1 52% (13/25), Top-5 96% (24/25)
Test Accuracy of livingroom: Top-1 68% (17/25), Top-5 96% (24/25)
Test Accuracy of restaurant: Top-1 48% (12/25), Top-5 92% (23/25)
Test Accuracy of subway: Top-1 80% (20/25), Top-5 100% (25/25)
Test Accuracy of warehouse: Top-1 88% (22/25), Top-5 100% (25/25)

 Overall Test Accuracy : Top-1 69% (173/250), Top-5 98% (245/250)
```

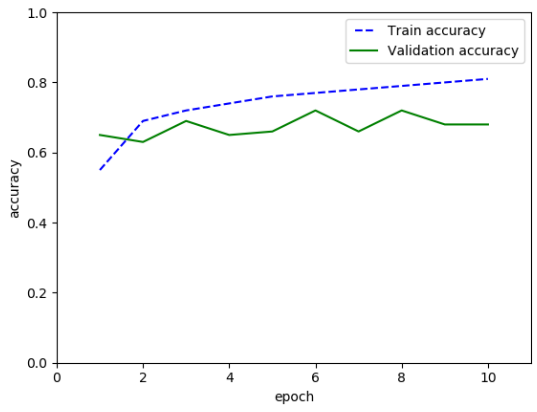


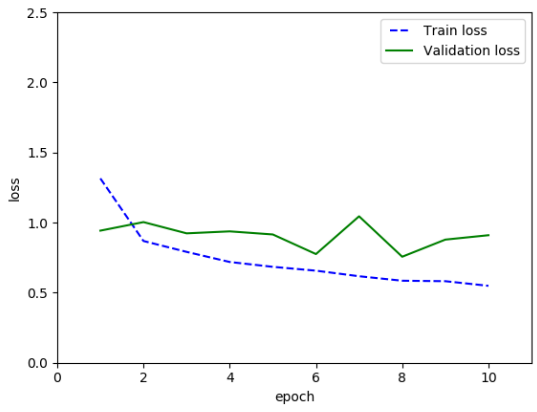


5 epochs
overall test accuracy%69, top5 test accuracy %96
learning rate = 0.001
epochs = 5
batch size = 25
number of workers = 2



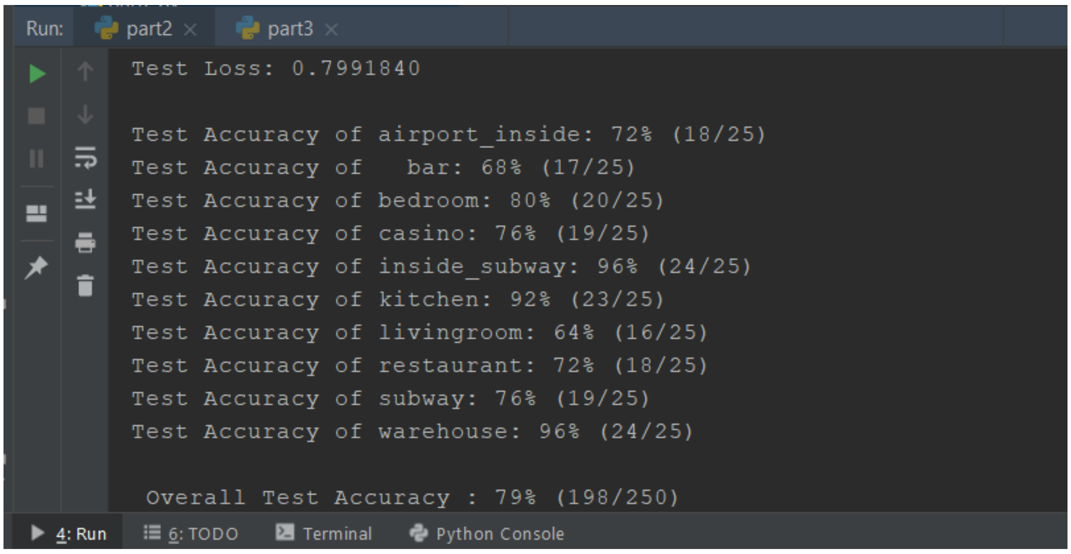
```
Test Accuracy of airport_inside: Top-1 76% (19/25), Top-5 100% (25/25)
Test Accuracy of   bar: Top-1 72% (18/25), Top-5 96% (24/25)
Test Accuracy of bedroom: Top-1 68% (17/25), Top-5 96% (24/25)
Test Accuracy of casino: Top-1 64% (16/25), Top-5 96% (24/25)
Test Accuracy of inside_subway: Top-1 92% (23/25), Top-5 100% (25/25)
Test Accuracy of kitchen: Top-1 72% (18/25), Top-5 96% (24/25)
Test Accuracy of livingroom: Top-1 64% (16/25), Top-5 100% (25/25)
Test Accuracy of restaurant: Top-1 64% (16/25), Top-5 100% (25/25)
Test Accuracy of subway: Top-1 60% (15/25), Top-5 100% (25/25)
Test Accuracy of warehouse: Top-1 88% (22/25), Top-5 100% (25/25)

 Overall Test Accuracy : Top-1 72% (180/250), Top-5 98% (246/250)
```



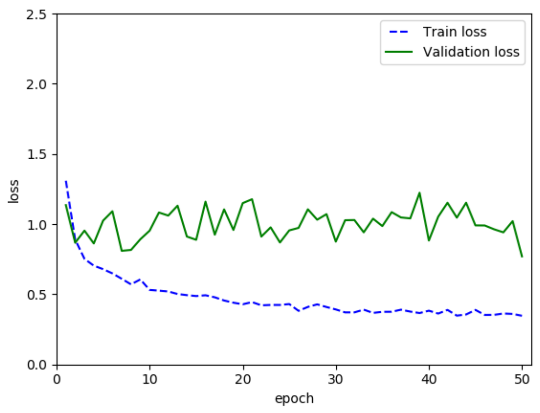


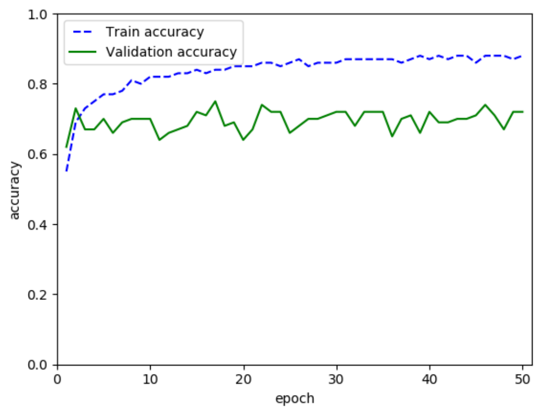
10 epochs

overall test accuracy%72, top5 test accuracy %98 , increased
learning rate = 0.001
batch size = 25
number of workers = 2



```
Test Loss: 0.7991840

Test Accuracy of airport_inside: 72% (18/25)
Test Accuracy of   bar: 68% (17/25)
Test Accuracy of bedroom: 80% (20/25)
Test Accuracy of casino: 76% (19/25)
Test Accuracy of inside_subway: 96% (24/25)
Test Accuracy of kitchen: 92% (23/25)
Test Accuracy of livingroom: 64% (16/25)
Test Accuracy of restaurant: 72% (18/25)
Test Accuracy of subway: 76% (19/25)
Test Accuracy of warehouse: 96% (24/25)

 Overall Test Accuracy : 79% (198/250)
```





50 epochs

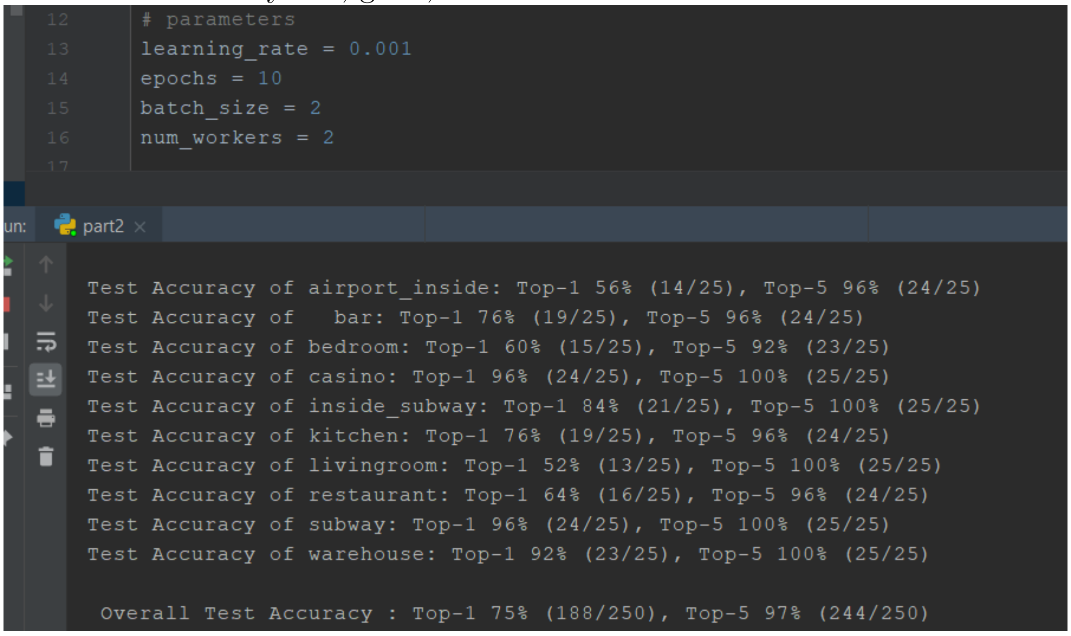
overall test accuracy%79, increased
learning rate = 0.001
batch size = 25
number of workers = 2
note: when i was trying with 50 epoch i haven't done it yet top5 accuracy, so it not shown. I did not have time do 50 epoch train again. But i think it will gives 99 100%.

Batch Size

Batch determines how much data is to be taken. If there is more, it will load more data, which will use more memory. When the Batch Size increase, more RAM is being used. But as the batch size increases, the accuracy often decreases. Because it learns with more data and the learning time is getting shorter. If the batch is reduced to you all the time, we can not say that the accuracy is improved. There must be an optimum range for this. 10 batch seems ideal.

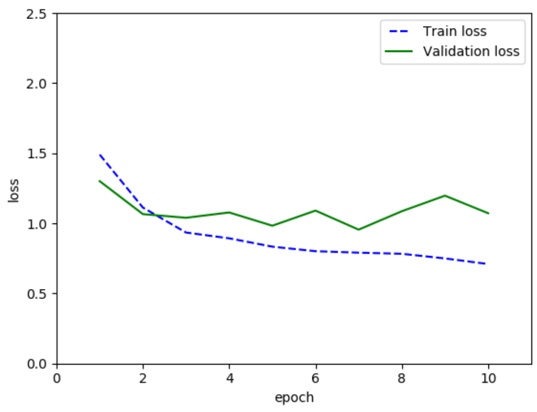
2 batch size:

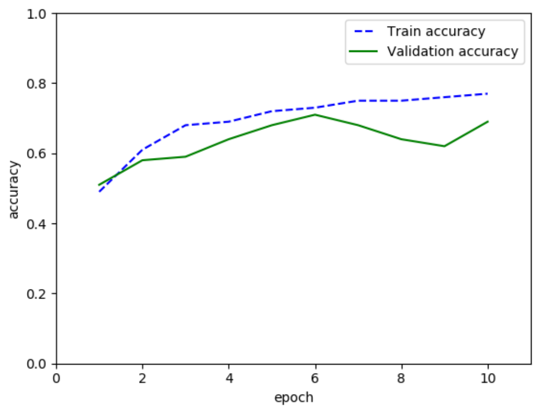
overall test accuracy%75, good, but slow



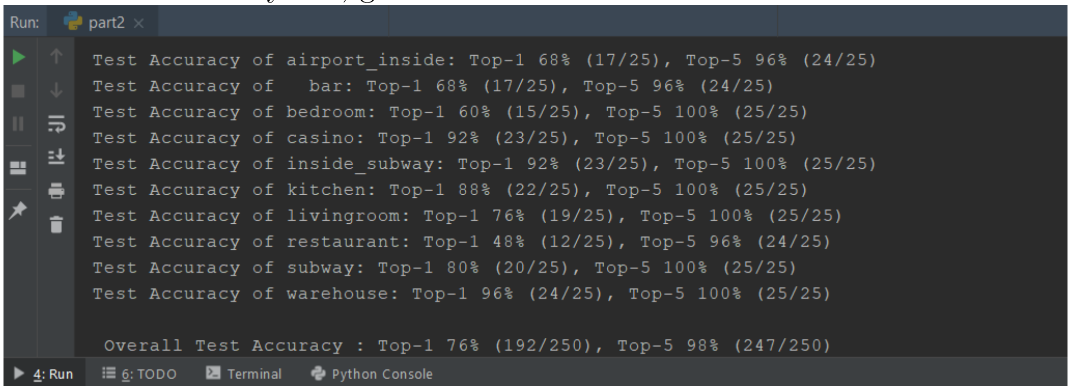
```
# parameters
learning_rate = 0.001
epochs = 10
batch_size = 2
num_workers = 2
```

```
Test Accuracy of airport_inside: Top-1 56% (14/25), Top-5 96% (24/25)
Test Accuracy of   bar: Top-1 76% (19/25), Top-5 96% (24/25)
Test Accuracy of bedroom: Top-1 60% (15/25), Top-5 92% (23/25)
Test Accuracy of casino: Top-1 96% (24/25), Top-5 100% (25/25)
Test Accuracy of inside_subway: Top-1 84% (21/25), Top-5 100% (25/25)
Test Accuracy of kitchen: Top-1 76% (19/25), Top-5 96% (24/25)
Test Accuracy of livingroom: Top-1 52% (13/25), Top-5 100% (25/25)
Test Accuracy of restaurant: Top-1 64% (16/25), Top-5 96% (24/25)
Test Accuracy of subway: Top-1 96% (24/25), Top-5 100% (25/25)
Test Accuracy of warehouse: Top-1 92% (23/25), Top-5 100% (25/25)

 Overall Test Accuracy : Top-1 75% (188/250), Top-5 97% (244/250)
```

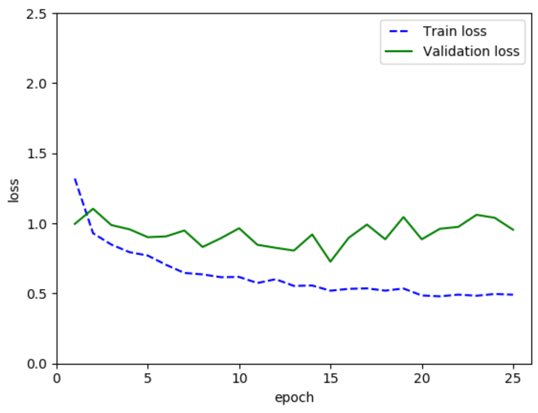


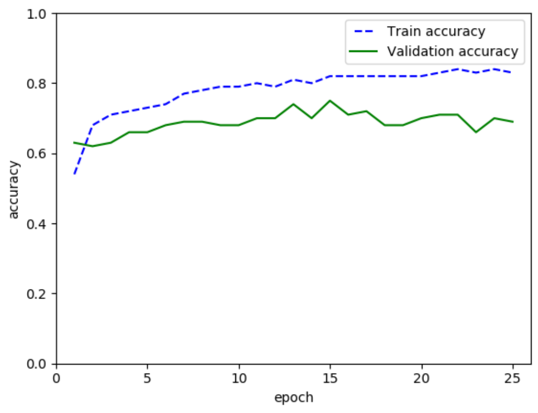


10 batch size:
overall test accuracy%76, good

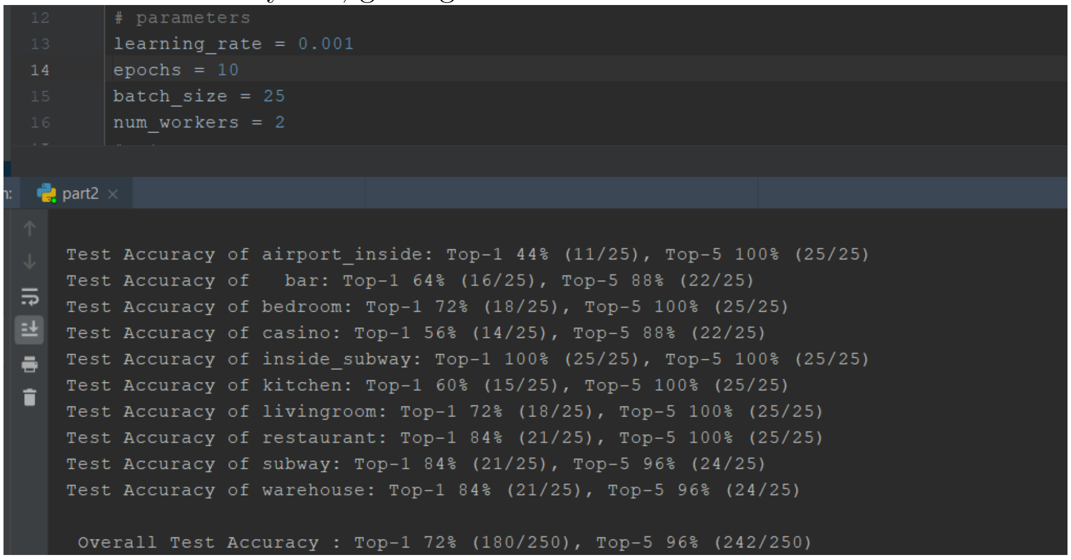


```
Test Accuracy of airport_inside: Top-1 68% (17/25), Top-5 96% (24/25)
Test Accuracy of   bar: Top-1 68% (17/25), Top-5 96% (24/25)
Test Accuracy of bedroom: Top-1 60% (15/25), Top-5 100% (25/25)
Test Accuracy of casino: Top-1 92% (23/25), Top-5 100% (25/25)
Test Accuracy of inside_subway: Top-1 92% (23/25), Top-5 100% (25/25)
Test Accuracy of kitchen: Top-1 88% (22/25), Top-5 100% (25/25)
Test Accuracy of livingroom: Top-1 76% (19/25), Top-5 100% (25/25)
Test Accuracy of restaurant: Top-1 48% (12/25), Top-5 96% (24/25)
Test Accuracy of subway: Top-1 80% (20/25), Top-5 100% (25/25)
Test Accuracy of warehouse: Top-1 96% (24/25), Top-5 100% (25/25)

 Overall Test Accuracy : Top-1 76% (192/250), Top-5 98% (247/250)
```





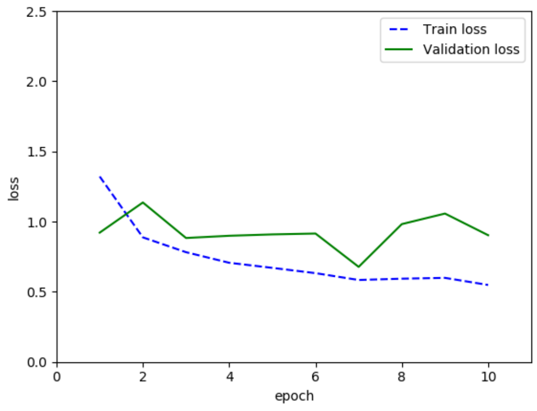
25 batch size:
overall test accuracy%72, getting little worse

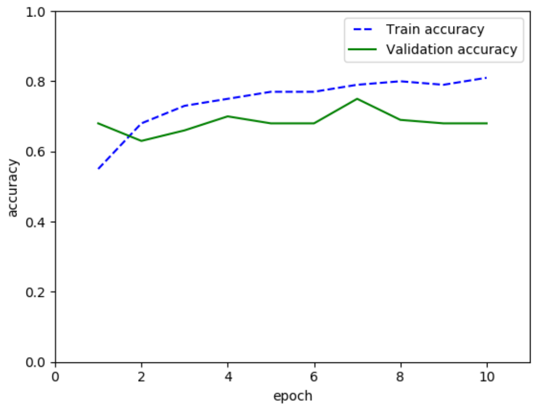


```
# parameters
learning_rate = 0.001
epochs = 10
batch_size = 25
num_workers = 2
```

part2

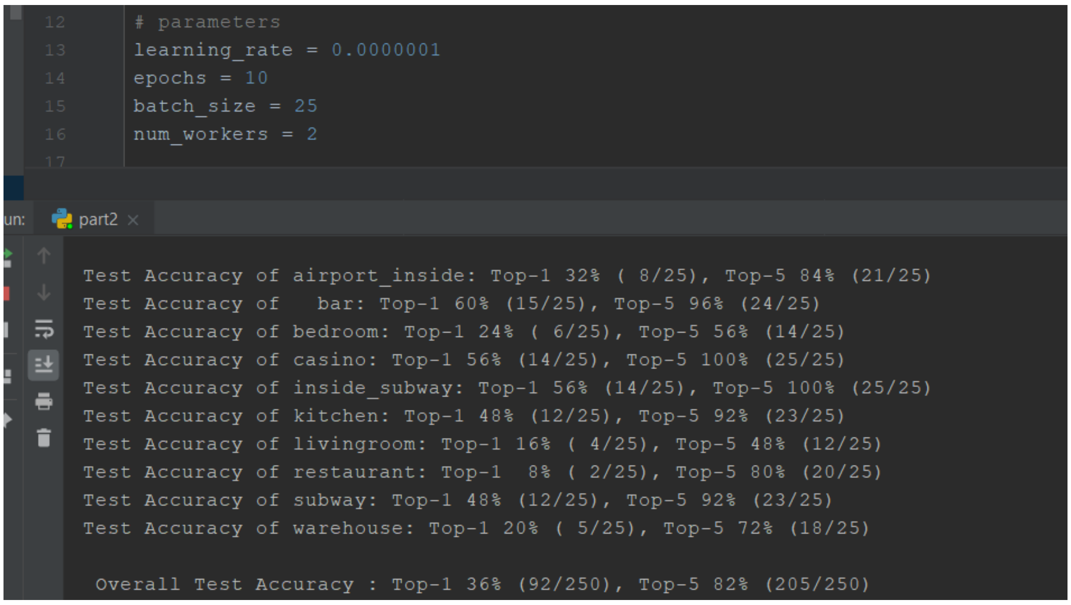
```
Test Accuracy of airport_inside: Top-1 44% (11/25), Top-5 100% (25/25)
Test Accuracy of   bar: Top-1 64% (16/25), Top-5 88% (22/25)
Test Accuracy of bedroom: Top-1 72% (18/25), Top-5 100% (25/25)
Test Accuracy of casino: Top-1 56% (14/25), Top-5 88% (22/25)
Test Accuracy of inside_subway: Top-1 100% (25/25), Top-5 100% (25/25)
Test Accuracy of kitchen: Top-1 60% (15/25), Top-5 100% (25/25)
Test Accuracy of livingroom: Top-1 72% (18/25), Top-5 100% (25/25)
Test Accuracy of restaurant: Top-1 84% (21/25), Top-5 100% (25/25)
Test Accuracy of subway: Top-1 84% (21/25), Top-5 96% (24/25)
Test Accuracy of warehouse: Top-1 84% (21/25), Top-5 96% (24/25)

 Overall Test Accuracy : Top-1 72% (180/250), Top-5 96% (242/250)
```





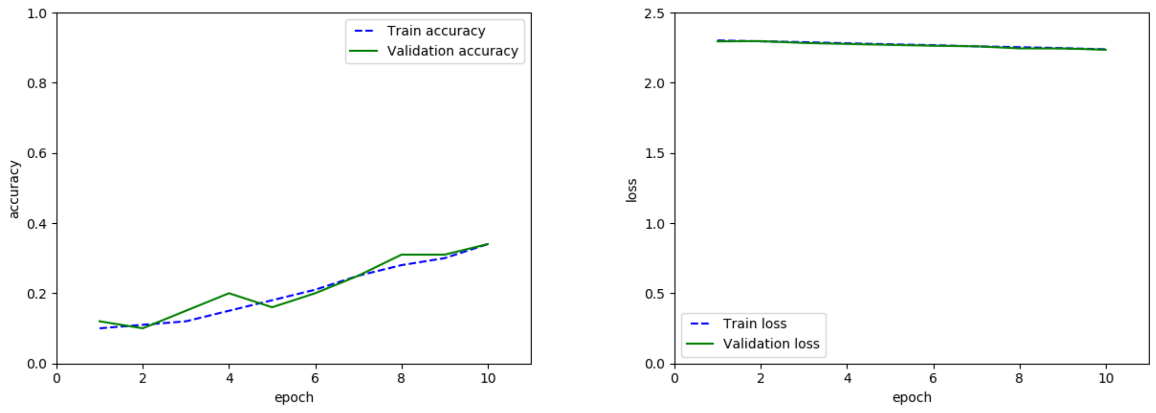
Learning Rate

The Learning Rate is one of the parameters that determines the weight change while the weightes are updated while the forward and backpropagation operations are performed on the Neural Network. If this parameter is too less, the loss value will be higher because each epoch will learn less. However, if the Learning Rate is high, also loss value is higher. With big learning rate we can not reach minimum loss because steps are too big.

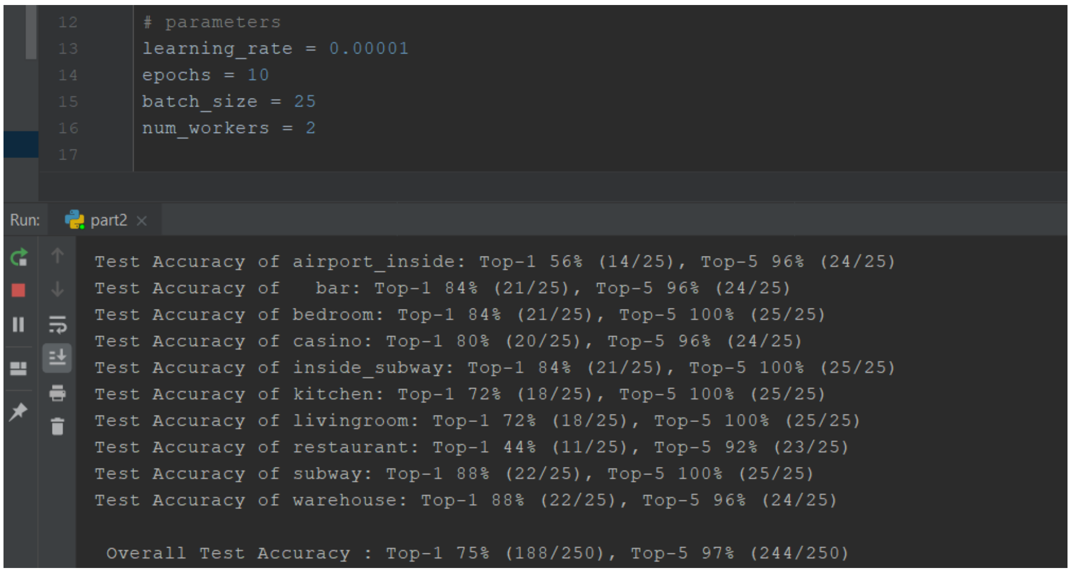


```
# parameters
learning_rate = 0.0000001
epochs = 10
batch_size = 25
num_workers = 2
```

```
Test Accuracy of airport_inside: Top-1 32% ( 8/25), Top-5 84% (21/25)
Test Accuracy of    bar: Top-1 60% (15/25), Top-5 96% (24/25)
Test Accuracy of bedroom: Top-1 24% ( 6/25), Top-5 56% (14/25)
Test Accuracy of casino: Top-1 56% (14/25), Top-5 100% (25/25)
Test Accuracy of inside_subway: Top-1 56% (14/25), Top-5 100% (25/25)
Test Accuracy of kitchen: Top-1 48% (12/25), Top-5 92% (23/25)
Test Accuracy of livingroom: Top-1 16% ( 4/25), Top-5 48% (12/25)
Test Accuracy of restaurant: Top-1  8% ( 2/25), Top-5 80% (20/25)
Test Accuracy of subway: Top-1 48% (12/25), Top-5 92% (23/25)
Test Accuracy of warehouse: Top-1 20% ( 5/25), Top-5 72% (18/25)

 Overall Test Accuracy : Top-1 36% (92/250), Top-5 82% (205/250)
```

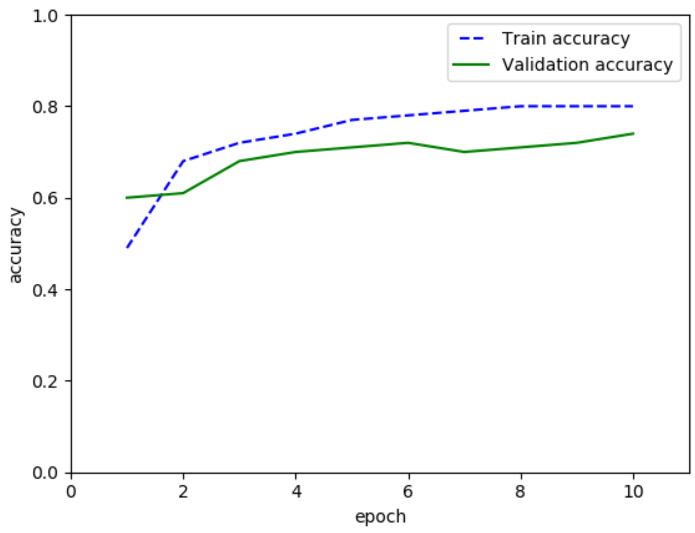


learning rate = 0.0000001
overall test accuracy%36. It gives bad accuracy because steps are too small, it learn less



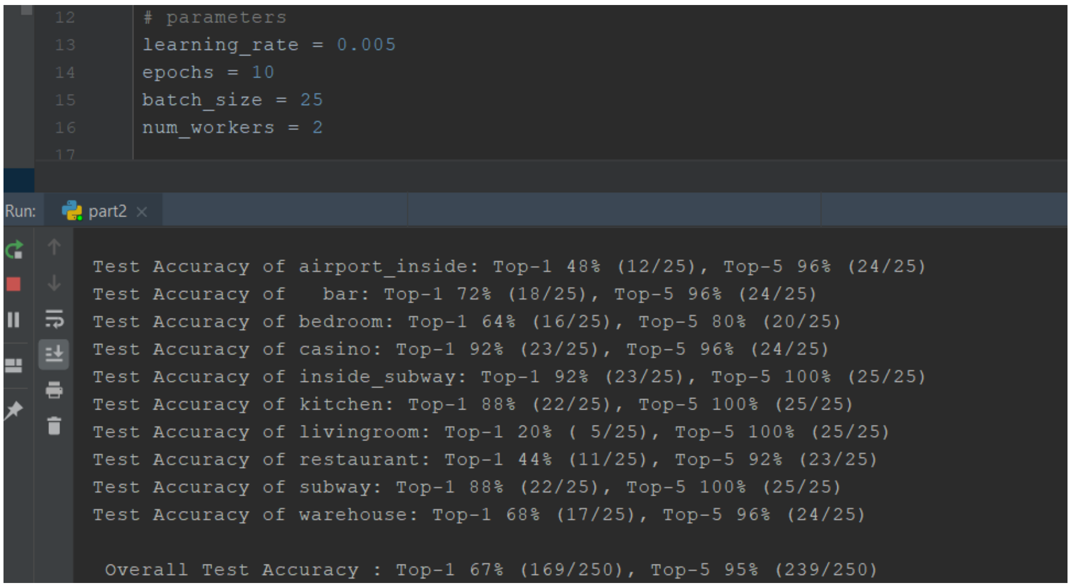
```
# parameters
learning_rate = 0.00001
epochs = 10
batch_size = 25
num_workers = 2
```

```
Test Accuracy of airport_inside: Top-1 56% (14/25), Top-5 96% (24/25)
Test Accuracy of   bar: Top-1 84% (21/25), Top-5 96% (24/25)
Test Accuracy of bedroom: Top-1 84% (21/25), Top-5 100% (25/25)
Test Accuracy of casino: Top-1 80% (20/25), Top-5 96% (24/25)
Test Accuracy of inside_subway: Top-1 84% (21/25), Top-5 100% (25/25)
Test Accuracy of kitchen: Top-1 72% (18/25), Top-5 100% (25/25)
Test Accuracy of livingroom: Top-1 72% (18/25), Top-5 100% (25/25)
Test Accuracy of restaurant: Top-1 44% (11/25), Top-5 92% (23/25)
Test Accuracy of subway: Top-1 88% (22/25), Top-5 100% (25/25)
Test Accuracy of warehouse: Top-1 88% (22/25), Top-5 96% (24/25)

 Overall Test Accuracy : Top-1 75% (188/250), Top-5 97% (244/250)
```



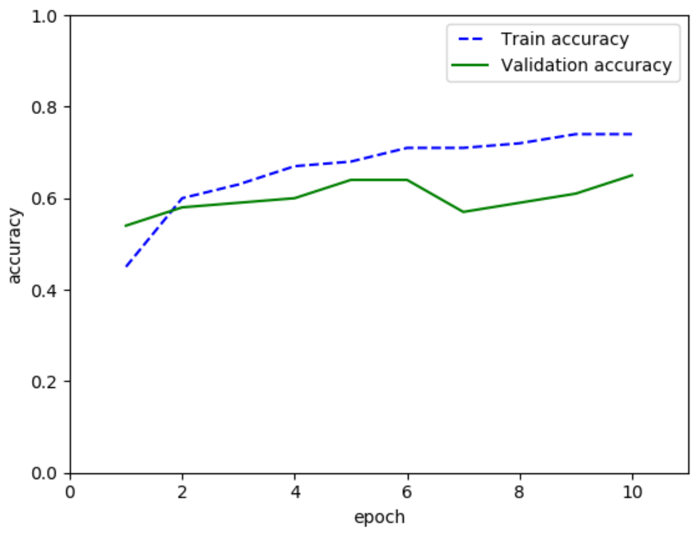
learning rate = 0.00001

overall test accuracy%75, good

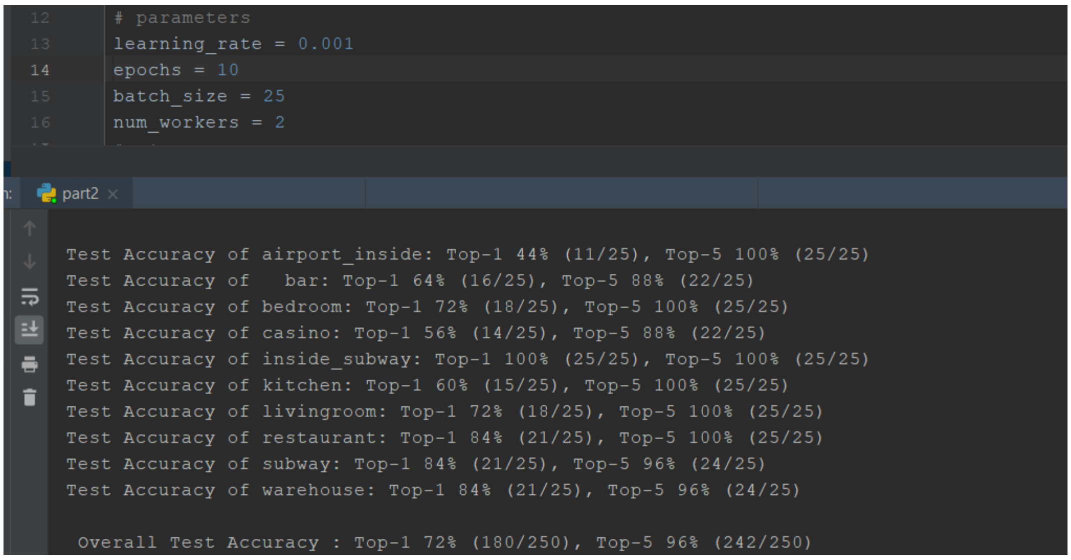


```
# parameters
learning_rate = 0.005
epochs = 10
batch_size = 25
num_workers = 2
```

```
Test Accuracy of airport_inside: Top-1 48% (12/25), Top-5 96% (24/25)
Test Accuracy of   bar: Top-1 72% (18/25), Top-5 96% (24/25)
Test Accuracy of bedroom: Top-1 64% (16/25), Top-5 80% (20/25)
Test Accuracy of casino: Top-1 92% (23/25), Top-5 96% (24/25)
Test Accuracy of inside_subway: Top-1 92% (23/25), Top-5 100% (25/25)
Test Accuracy of kitchen: Top-1 88% (22/25), Top-5 100% (25/25)
Test Accuracy of livingroom: Top-1 20% ( 5/25), Top-5 100% (25/25)
Test Accuracy of restaurant: Top-1 44% (11/25), Top-5 92% (23/25)
Test Accuracy of subway: Top-1 88% (22/25), Top-5 100% (25/25)
Test Accuracy of warehouse: Top-1 68% (17/25), Top-5 96% (24/25)

 Overall Test Accuracy : Top-1 67% (169/250), Top-5 95% (239/250)
```



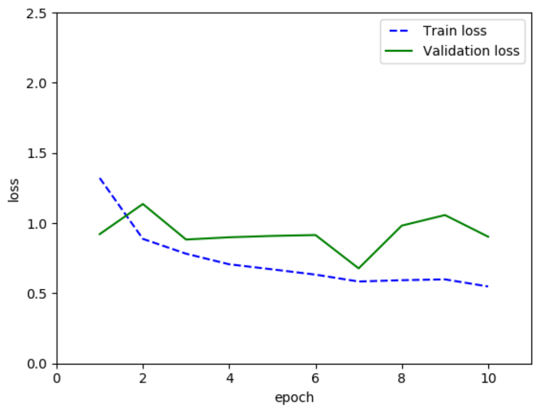
learning rate = 0.005 overall test accuracy%67, getting worse

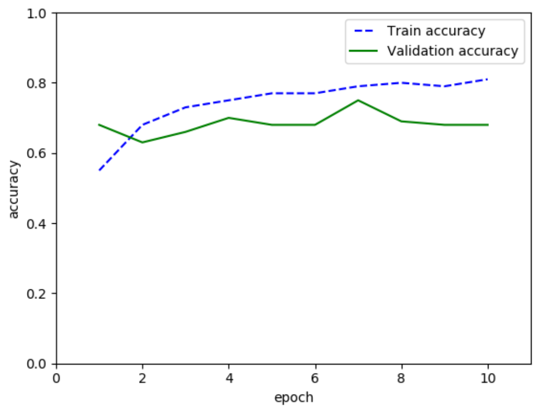


```
# parameters
learning_rate = 0.001
epochs = 10
batch_size = 25
num_workers = 2
```

part2

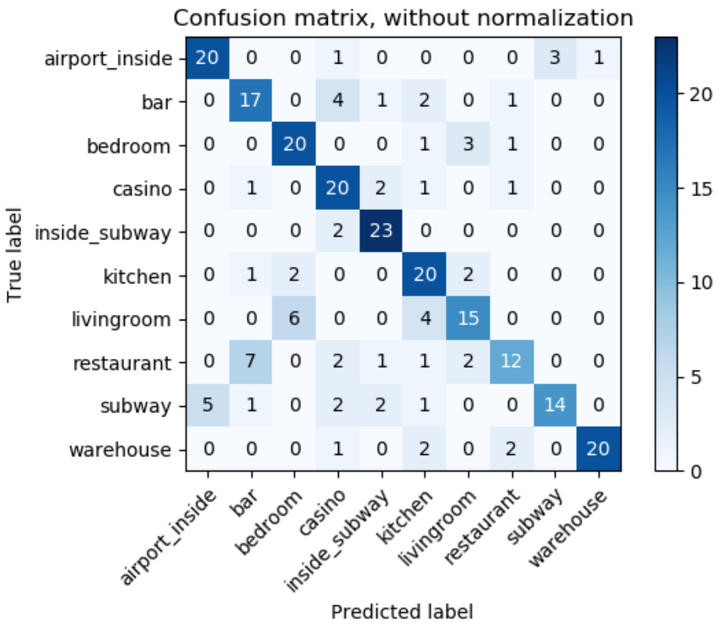
```
Test Accuracy of airport_inside: Top-1 44% (11/25), Top-5 100% (25/25)
Test Accuracy of   bar: Top-1 64% (16/25), Top-5 88% (22/25)
Test Accuracy of bedroom: Top-1 72% (18/25), Top-5 100% (25/25)
Test Accuracy of casino: Top-1 56% (14/25), Top-5 88% (22/25)
Test Accuracy of inside_subway: Top-1 100% (25/25), Top-5 100% (25/25)
Test Accuracy of kitchen: Top-1 60% (15/25), Top-5 100% (25/25)
Test Accuracy of livingroom: Top-1 72% (18/25), Top-5 100% (25/25)
Test Accuracy of restaurant: Top-1 84% (21/25), Top-5 100% (25/25)
Test Accuracy of subway: Top-1 84% (21/25), Top-5 96% (24/25)
Test Accuracy of warehouse: Top-1 84% (21/25), Top-5 96% (24/25)

 Overall Test Accuracy : Top-1 72% (180/250), Top-5 96% (242/250)
```



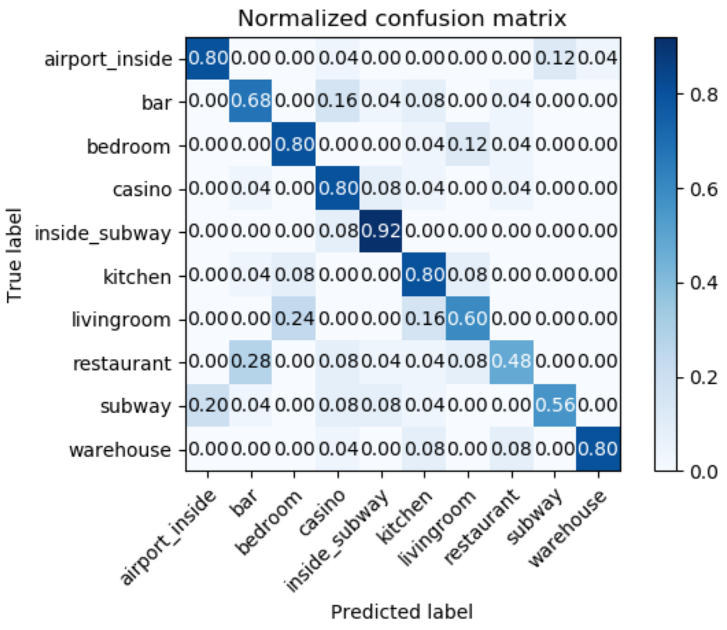


learning rate = 0.001
overall test accuracy%72, good also

3.3 PART 3:

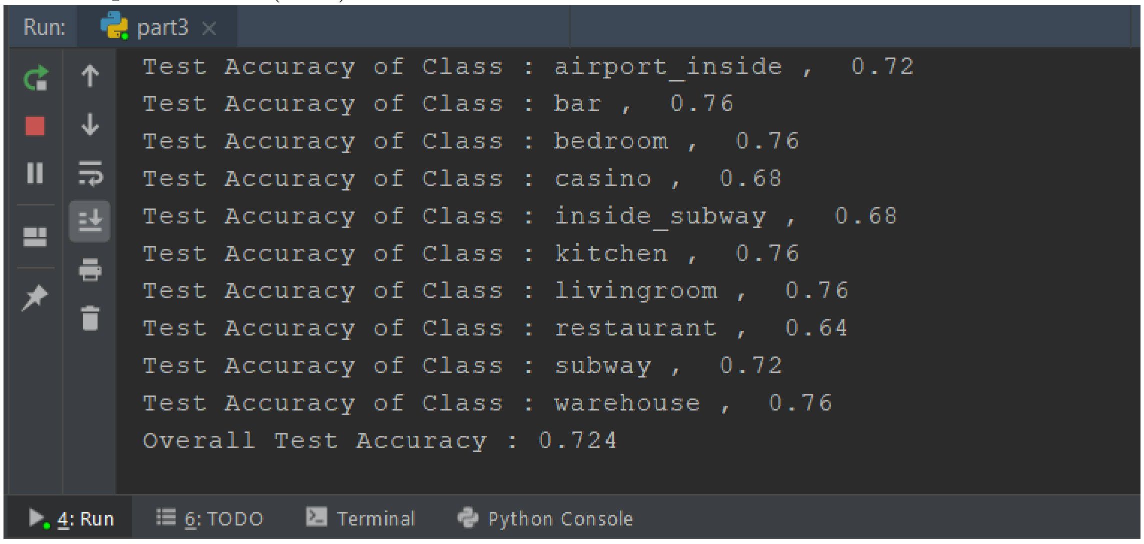


confusion matrix



normalized confusion matrix

By looking the confusion matrix, network bad at restaurant, subway, livingroom classes. Very good at inside subway class. Network confused in restaurant with bar a lot(%28). Also livingroom confused with bedroom a lot(%24). Subway confused with airport inside (%20) etc.



```
Test Accuracy of Class : airport_inside ,  0.72
Test Accuracy of Class : bar ,  0.76
Test Accuracy of Class : bedroom ,  0.76
Test Accuracy of Class : casino ,  0.68
Test Accuracy of Class : inside_subway ,  0.68
Test Accuracy of Class : kitchen ,  0.76
Test Accuracy of Class : livingroom ,  0.76
Test Accuracy of Class : restaurant ,  0.64
Test Accuracy of Class : subway ,  0.72
Test Accuracy of Class : warehouse ,  0.76
Overall Test Accuracy : 0.724
```

overall test accuracy %72.4
batch size = 25
num workers = 4
As seen in the accuracy, result is better then part1. I did same operations in part3 but i use pretrained of part2 instead of pretrained vgg16. Model at part2 was trained and saved and i loaded it. That's why i get better result, because the model i used improved in part2.

4 Conclusion

Results are satisfactory as well. I did the expected CNN and Fine-Tuning operations and compared the effects of different epoch numbers. I also tried to interpret the results by comparing different learning rates with different batch sizes. Also i showed the each part. I can not see any overfit or underfit between these results. Train Loss and Accuracy values are not bad. Top5 accuracy reached about 96% 98%, it is good result, i did not expect that. But after a certain train it may be normal to find one out of the top 5 possibilities. So this is normal. The experiment was difficult, i failed too much nearly everywhere but i learned a lot from the experiment.