Assignment 3

Name(s): Pallaw Kumar and Neha Jain
NetID(s): pallawk2 and nehaj4

Part 1: Self-supervised Learning on CIFAR10

1) Rotation training

Report the hyperparameters you used to train your model. Discuss any particular implementation choices which caused significant performance increases.

Optimizer: Adam Loss: Cross Entropy Loss Learning Rate: 0.01 Decay Epochs: 15
Number of epochs: 45

There is a significant change in test accuracy when the optimizer is changed from SGD to Adam. With SGD optimizer testing accuracy on the 10000 test images is 66.19 and the average loss is 0.819. on the other hand, the Adam optimizer gives testing accuracy on the 10000 test images of 79.04 and an average loss of 0.527.

2) Fine-tuning late layers
Report the hyperparameters you used to fine-tune your model. Compare the performance between pre-trained model and randomly initialized model.

Fine-tuning on the pre-trained model ResNet18 keeping frozen all previous layers except for ‘layer4’ and ‘fc’ layer:

Optimizer: Adam
Loss: Cross Entropy Loss
Learning Rate: 0.01,
Number of epochs =20,
decay epochs=10

Accuracy of the network on the 10000 test images: 62.86 %
The average loss on the 10000 test images: 1.055

Fine-tunning on the randomly initialized model ResNet18 keeping frozen all previous layers except for ‘layer4’ and ‘fc’ layer:

Optimizer: Adam
Loss: Cross Entropy Loss Learning
Rate: 0.01,

learning rate: 0.001,
Number of epochs =20,
decay epochs=10

Accuracy of the network on the 10000 test images: 45.90 %
The average loss on the 10000 test images: 1.525

3) Fully supervised learning

Report the hyperparameters you used to fine-tune your model. Compare the performance between pre-trained model and randomly initialized model. Discuss anything you find interesting comparing fine-tuning the late layers only in section (2) and fine-tuning the whole model in section (3).

Load the pre-trained ResNet18 model and re-train the whole model on the classification task:

Optimizer: Adam
Loss: Cross Entropy Loss
Learning Rate: 0.01,
Number of epochs =20,
decay epochs=10
Accuracy of the network on the 10000 test images: 83.46 %
The average loss on the 10000 test images: 0.491

Randomly initialize a ResNet18 model and re-train the whole model on the classification task:

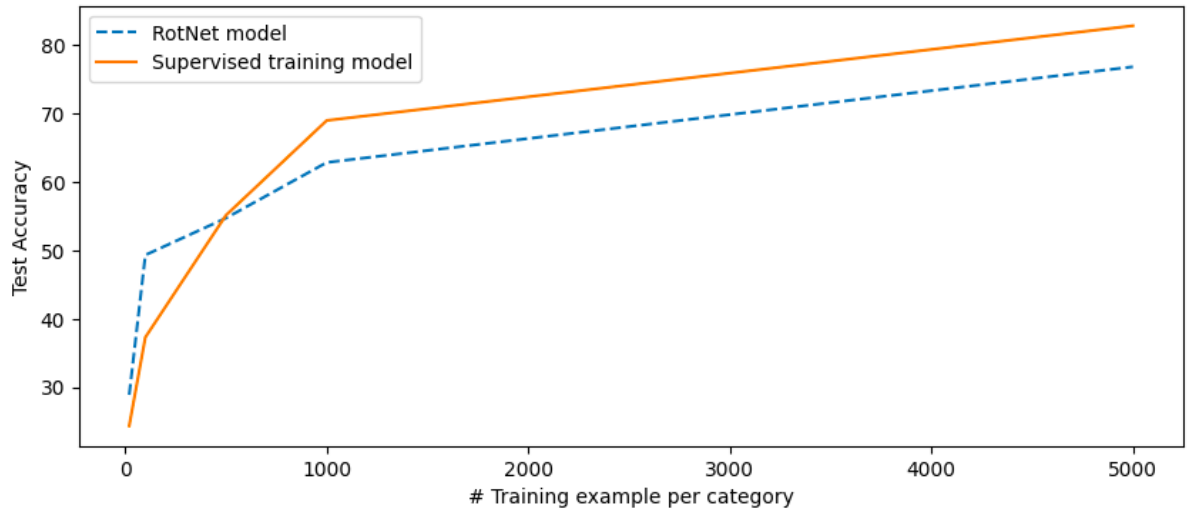
Optimizer: Adam
Loss: Cross Entropy Loss
Learning Rate: 0.01,
Number of epochs =20,
decay epochs=10
Accuracy of the network on the 10000 test images: 82.36 %
The average loss on the 10000 test images: 0.521

In section two when the pre-trained model is used for fine-tuning the late layers on the classification task gives a testing accuracy of 62.86 % and an average loss of 1.055 while for randomly initialized model task gives 45.90 % and an average loss of 1.525. which means the model learns much better when a pre-trained model is used for the classification task due to the rotation task's learned weights' ability to recognize several important features needed for the classification problem. In the case of fine-tuning the whole model with a pre-trained model and randomly initialized model performs better than section two models. From this we can understand that if we have a smaller dataset then we should use self-supervised learning for learning important features then we should use this pre-trained model as a transfer learning for the classification task.

4) Extra credit

Part 1:

Number of training examples per category	RotNet model	Supervised training
20	**24.34**	**22.06**
100	**37.29**	**34.24**
500	**55.14**	**53.32**
1000	**69.0**	**64.92**
5000	**82.85**	**84.1**



we attempted to recreate the 5(b) graph. The number of training examples per category (in the subset of training data used to train both the supervised CIFAR10 model and a RotNet model with the final layers tuned on CIFAR10) is represented by the Xaxis of the graph. The test accuracy is shown on the Y-axis. The graph below was made using five training data subsets of 20,100, 500, 1000, and 5000 examples per category, respectively. We can observe that the RotNet model with final layers based on CIFAR10 outperforms the supervised CIFAR10 model in cases with fewer examples per category.

Part 2: A. We have used ResNet50 for the rotation prediction task. Using the Adam optimizer with initial learning rate= 0.01 and decay after every 15 epochs and saved the model and then used the saved model to train the model further with the Adam optimizer. Total epochs = 255 epochs. Finally result:

Accuracy of the network on the 10000 test images: 80.17 % (With resnet 18 accuracy of the network on the 10000 test images: 79.04 %)
The average loss on the 10000 test images: was 0.511 (Resnet18 avg loss on 1000 test images: 0.527).
So, the accuracy of RotNet using resnet50 is more than when using resnet18.

B. Used the above trained RotNet model using Resnet50 for transfer to supervised CIFAR10 classification task. Used Adam optimizer with decay in the learning rate, and trained for 30 epochs.
Accuracy of the network on the 10000 test images: 84.10 % (With RotNet using Resnet 18 as pre-trained, the accuracy of the network on the 10000 test images: 83.46%)
The average loss on the 10000 test images: 0.473 (With RotNet using Resnet18 as pre-trained, avg loss on 1000 test image: 0.491). So, the accuracy of RotNet using resnet50 for pre-train for CIFAR10 classification task is more than when using RotNet using Resnet18 for pretrain for CIFAR10 classification task.

Part 3: We loaded the imagenette dataset(of 160px) and calculated the mean and standard deviation of the train data for normalization transformation mean = [0.4666, 0.4588, 0.4301] and Standard deviations = [0.2330, 0.2269, 0.2348]. Used a batch size of 16, and adam optimizer with initial learning rate = with decay after few epochs, ran the mode for 40 epochs. Final output:
Accuracy of the network on the 10000 test images: 58.70 %
Average loss on the 10000 test images: 0.970

Part-2: Object Detection by YOLO

1. My best mAP value on Kaggle :1-mAP(i.e. Score on Kaggle) = 0.48466 (Team name: Ninja4)
2. Did you upload final CSV file on Kaggle: Yes
3. My final loss value : training loss: 1.925 and best test loss : 2.97
4. What did not work in my code(if anything): Initially I was using a threshold of 0.1 in non-max suppression, but I was getting many bounding boxes, so I changed it to 0.5. The results are below for both of them.
5. Sample Images from my detector from PASCAL VOC:

With nms threshold = 0.1

With nms threshold = 0.5

Extra Credit for YOLO :

1. I produced a video by running the detector (using best weights saved in best_detector.pth).
2. Following steps are performed to get the output:

a. First, I used 500 frames of the 'SNL Digital Short- YOLO - SNL.mp4' video and stored those frames in a folder(input_frames)
b. Passed each frame saved in the input_frames folder through the network and predicted the objects, their class, and confidence (nms threshold = 0.35) on the input frame and stored it in a folder named output_frames.
c. Lastly, created the video named video_nehaj4_pallawk2.mp4 using the frames in the output_frames folder.