16824 HW1 Report

Zhening Yang

March 2, 2018

1 Task 0: MNIST 10-digit classification in TensorFlow

After training the model in 00_mnist.py for 20k iterations, the training loss decrease to 0.103 and the accuracy of validation set is 96.9% in the MNIST digits dataset.

After extending iterations to 30K, the loss is keeping decrease to 0.0743, the accuracy slightly increase to 97.7%. It should be caused by overfitting.

The following two figures show the change of training loss and validation accuracy in 20K iterations.

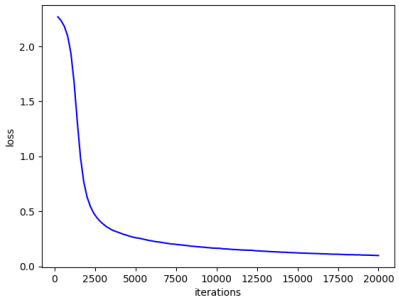


Figure 1: Training loss

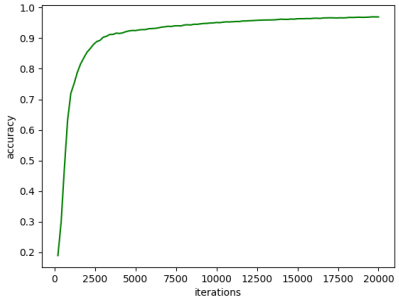


Figure 2: Validation accuracy

2 Task 1: 2-layer network for PASCAL multi-label classification

With the two-layer model, the following images show the training loss and mean average precision in 1K iterations of PASCAL 2007 dataset.

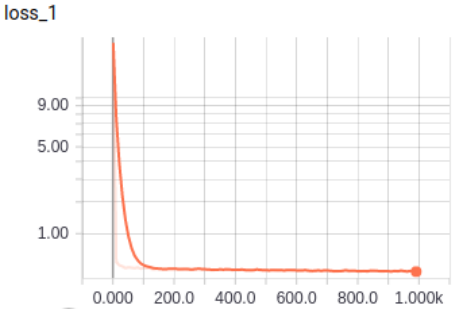


Figure 3: Training loss curve of two-layer network

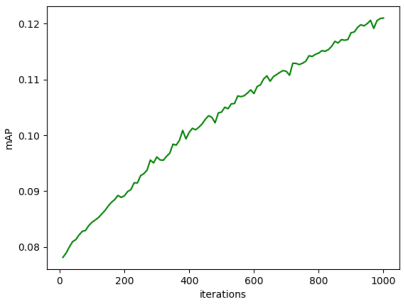


Figure 4: mAP curve of two-layer network

The loss decreased slowly in the training process and final mAP is 12.1%.

3 Task 2: AlexNet for PASCAL classification

After modifying previous network to AlexNet, changing the initialization methods of weights and optimizer to SGD + Momentum. The following images show the training loss and mean average precision in 40K iterations of PASCAL 2007 dataset with data augmentation.

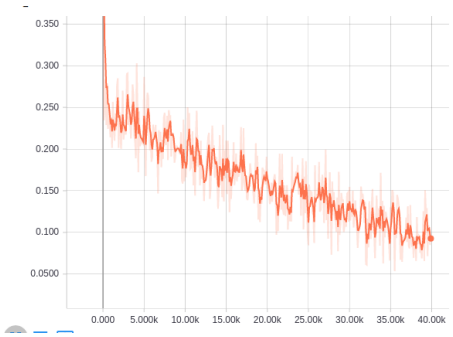


Figure 5: Training loss curve of AlexNet

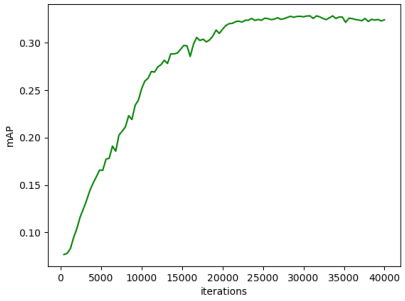


Figure 6: mAP curve of AlexNet

As we can see, the loss keep decreasing in a normal speed, due to small batch size (10), the curve looks fluctuating. After mAP reaching 32%, the accuracy curve seems stop but loss keeping decreasing which may caused by overfitting.

4 Task 3: VGG-16 for PASCAL classification

After modifying previous network to VGG-16[2]. The following images show the training loss and mean average precision in 40K iterations of PASCAL 2007 dataset.

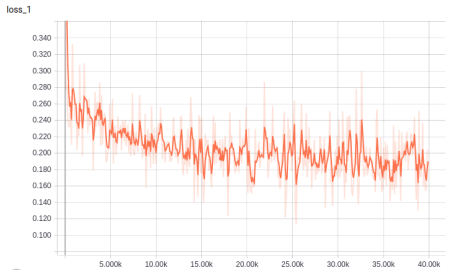


Figure 7: Training loss curve of VGG-16

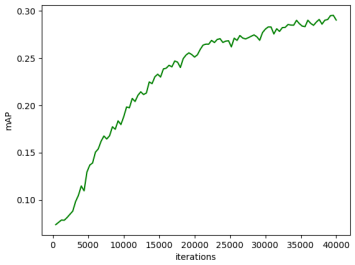


Figure 8: mAP curve of VGG-16

Compared to AlexNet, the decreasing trend of loss when training VGG-16 is not so good, and the mAP is less than AlexNet. One possible reason for this is the choice of initialization method, I tried Gaussian initialization first but mAP decreased in the process of training. One possible reason is gradient vanish, gradients in the previous layers could not affect the later weights because of bad weight initialization, which could lead to severe overfitting in the latter full connected layers.

Figure 9 shows the whole network graph of VGG-16, Figure 10 and Figure 11 are some samples of gradient histograms and training images. Using exponential decay learning rate to training the model. Figure 12 is the learning rate curve.

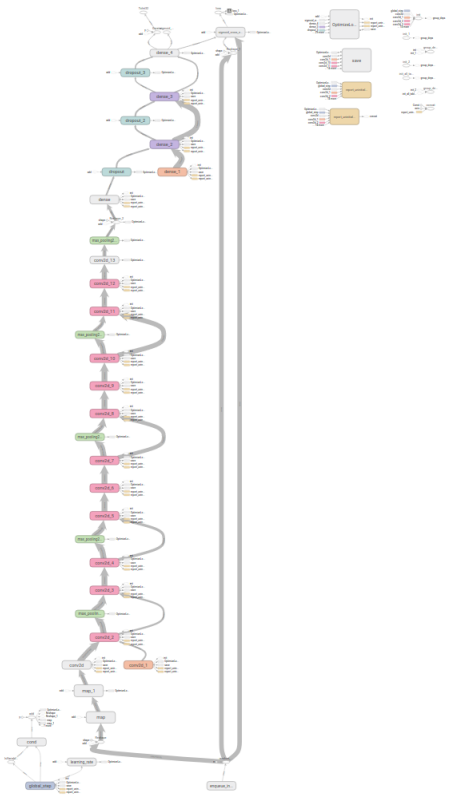


Figure 9: Network graph of VGG-16

Figure 10: Samples of VGG-16 gradient histogram

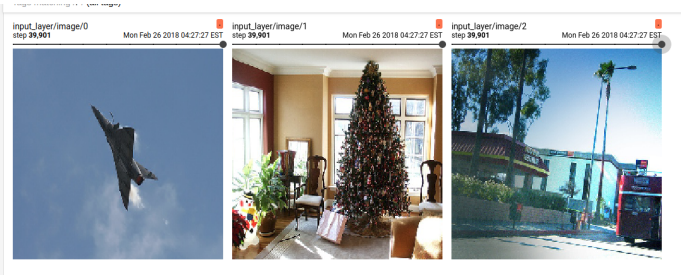


Figure 11: Samples of VGG-16 training images

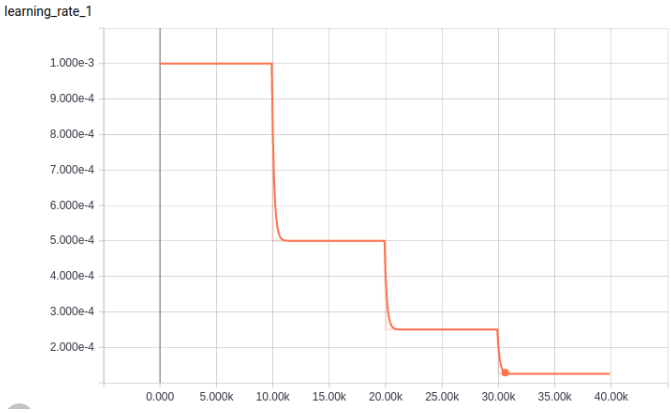


Figure 12: Learning rate in 40K iterations

5 Task 4: Fine-tuning VGG-16 from ImageNet

After using the pre-trained VGG-16 for fine tuning. The following images show the training loss and final mean average precision after 4K iterations. The final mAP is 65.3

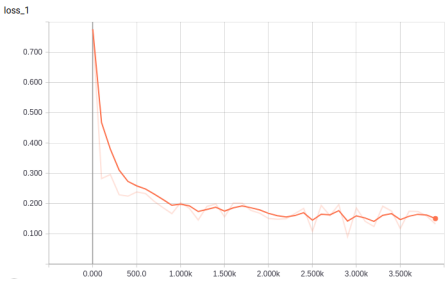


Figure 13: Training loss of fine-tuning VGG-16

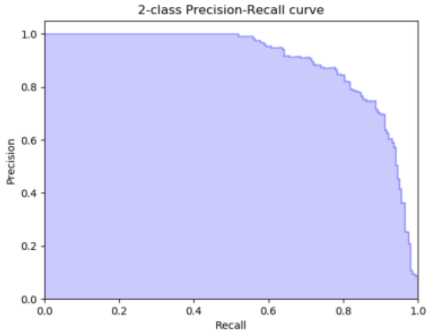
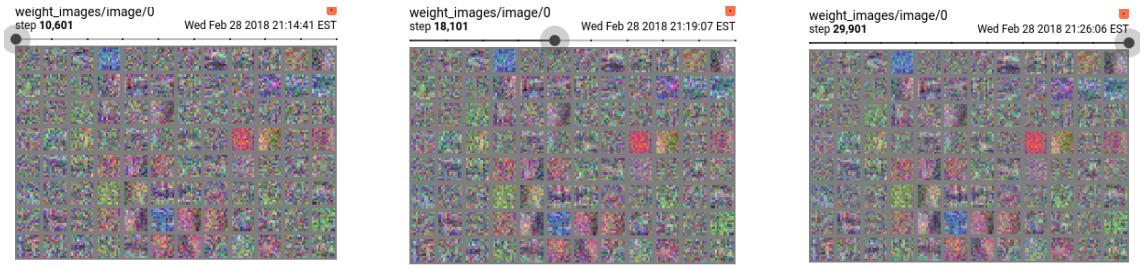


Figure 14: Final mAP of fine-tuning VGG-16

6 Task 5: Analysis

A series of excavation could be done to better understand the training result. First, by visualizing the weights of first convolution layer of AlexNet, the most notable thing is that the filter become more and more smooth in the training process.



(a) 10k iteration (b) 20k iteration (c) 30k iteration

Figure 15: Conv1 weights through 30k iterations

With trained features, we could do some interesting stuff like find the nearest neighbour of one image among whole dataset. Table1 shows the nearest neighbours of ten different class images with features from layers like Alex pool5, Alex fc7, VGG pool5 and VGG fc7 respectively. From the results we could notice that, for network which is not deep enough, such as AlexNet, it is easy for it to rely on some common features like color. However, for deep VGG model, it usually has better generalization ability to extract more abstract features. For example, in the table's 8th row(person), the target image is a person drinking, the AlexNet could only find neighbours with people wearing blue cloths, but the VGGNet could find images with people also drinking.

Since we have valuable features, it is also useful to know whether these features could classify objects efficiently. tSNE visualization is helpful for us to observe how well the images could be classified by these features. Figure 16 shows the tSNE visualization of VGG fc7 layer.

What's more, combining these results with the final predict accuracy of every class. Obviously, some classes are harder to be recognized than others. For example, the potted-plant and bottle, I think these objects are hard for recognition for two reasons. First, the number of images contains these class are fewer than other common classes. In the train set, there are more than 2000 images contain person, which has the best prediction accuracy among all classes, but there are only 244 positive samples of bottles, it is hard for network to generalize from small amount of data. second, these objects usually not the only object in the images or occur with random numbers. It is hard for network to extract unique or general features from it.

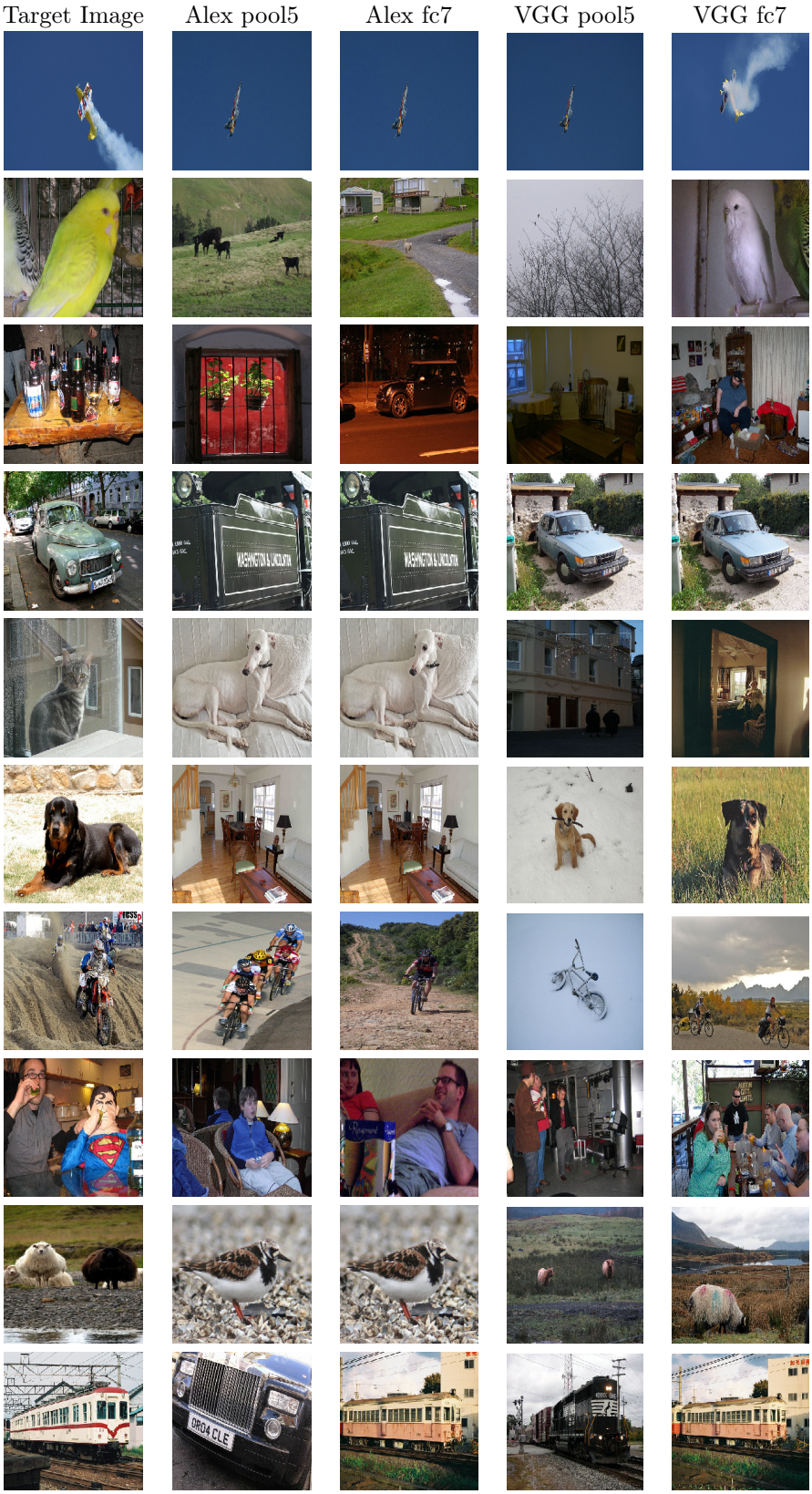


Table 1: Nearest neighbour based on pool5/fc7 features of Alex/VGG-16 model

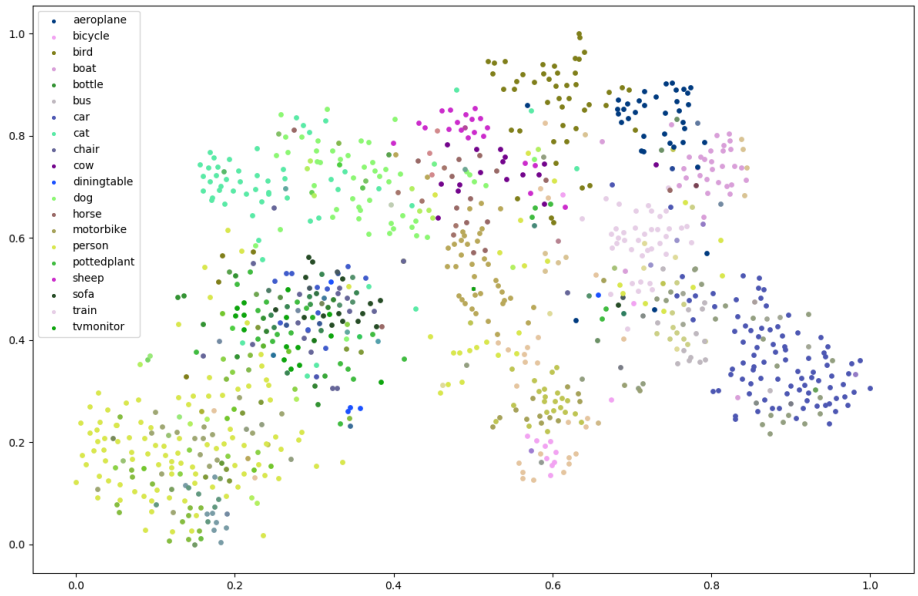


Figure 16: tSNE visualization of VGG fc7 layer

Class	Prediction accuracy
aeroplane	0.878594653874
bicycle	0.730142683739
bird	0.824085666621
boat	0.801139878846
bottle	0.272340504879
bus	0.696550438127
car	0.874913634934
cat	0.818396078173
chair	0.520147482253
cow	0.578232957741
diningtable	0.442927859475
dog	0.756717385391
horse	0.766953049491
motorbike	0.731723086299
person	0.931363104216
pottedplant	0.262737066309
sheep	0.38094583843
sofa	0.480839009898
train	0.855766789706
tvmonitor	0.402399475103

Table 2: Final predict accuracy of every class

7 Task 6: Improve the classification performance

After trying the mixup[2] technique with different parameter to do extra data automation, unfortunately I did not find observable improvement compares to orginal AlexNet baseline. The

following images show the training loss and mean average precision in 40K iterations of PASCAL 2007 dataset with mixup.

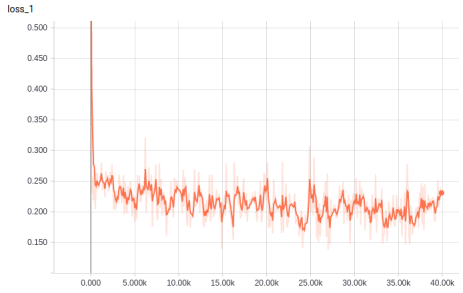


Figure 17: Training loss curve of mixup AlexNet

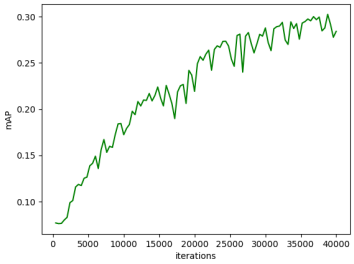


Figure 18: mAP curve of mixup AlexNet

References

[1] Karen Simonyan, Andrew Zisserman *Very Deep Convolutional Networks for Large-Scale Image Recognition*. arXiv:1409.1556

[2] Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, David Lopez-Paz *mixup: Beyond Empirical Risk Minimization*. arXiv:1710.09412