Oday Bani Ahmad

A20344588

[Question 1]

1. A regular CNN where the number of filters in each layer increases as the depth of the

network grows i.e., the Lth layer will have more filters than the (L-1)th layer.

The best of my trial was the following

* SGD , Batch size = 32 , epochs = 25 , val\_loss: 0.0228 - Val accuracy: 0.9946

2. An inverted CNN where the number of filters in each layer decreases as the depth of the network grows i.e., the Lth layer will have less filters than the (L-1)th layer.

The best of my trial was the following

* SGD , epochs = 1 , batch size = 128 , val\_loss: 0.1149 - val\_accuracy: 0.9825

3. An hour-glass shaped CNN where the number of filters will increase till the Lth layer and reduce afterward.

The best of my trial was the following

* RMSProp , epochs = 25 , batch size = 256 , val\_loss: 0.0336 - val\_accuracy: 0.9907

MY notes:

* By increasing the epochs, the model accuracy will increase and with a high number of epochs like 25, the accuracy will increase to the epochs 12-15 and then it decreases little on the last epochs.
* On the first model and the second of this question SGD optimizer performed the best. On the third model, RMSProp performed the best.
* With a smaller batch size the value of loss will be less.
* Learning rate doesn’t affect a lot at this data in these models and that might be because I kept using Lr (.001 - .00001).
* Using this model structure with 10 Conv2d layers enhances the model accuracy and minimizes the value of the loss.

[Question 2]

1. What is the effect of learning rate on the training process? Which performed best?

*Lr controls how quickly or slowly a neural network model learns a problem. Moreover, the amount that the weights are updated during training is referred to as the step size.* *in my application there was, no significance affects but the range of values to consider for the learning rate is less than 1.0 and greater than 10^-6. A traditional default value for the learning rate is 0.1 or 0.01, and this may represent a good starting point on your problem.*

2. What is the effect of batch size on the training process? Which performed best?

*Batch size controls the accuracy of the estimate of the error gradient when training neural networks. When I used a variety of batch size i noticed that with the smaller batch size the value of loss will be less*

3. Try different hyperparameters to obtain the best accuracy on the test set. What is your

best performance and what were the hyperparameters?

My best output:

learning rate=0.00001, batch = 16 acc =: 0.5507

Using Adam with the following set of epochs = 25 and the following hyperparameters:

1. acc = 0.5495
2. learning rate=0.001 acc =:0.5429
3. learning rate=0.0001 acc =:0.5488
4. learning rate=0.0001, batch = 512 acc =: 0.5484
5. learning rate=0.0001, batch = 256 acc =: 0.5477
6. learning rate=0.0001, batch = 1028 acc =: 0.5471 **the fastest**
7. learning rate=0.0001, batch = 32 acc =: 0.5462
8. learning rate=0.00001, batch = 16 acc =: 0.5507

4. Implement an equivalent feed forward network for the same task with each hidden layer containing the same number of neurons as the number of filters in each convolution layer. Use the ‘Adam’ optimizer to train your network on the CIFAR-10 dataset for a fixed set of 25 epochs. Compare its performance with your LeNet implementation based on the following questions:

a. What is its performance?

*best performance with batch size 512 as the acc = 0.3264 per my trail.*

b. How many parameters are there in this network compared to the LeNet

implementation? Are they worth it?

*At this model the Total params: 872,510 Trainable params: 872,510 non-trainable params: 0*

*At LeNet Total params: 92,246 Trainable params: 92,246 non-trainable params: 0 it is not worth to have such and huge amount of params especially that the accuracy is less in these feedforward network.*

[Question 3]

1. What are the dimensions of the input and the kernel (or filter)? How many parameters

are there in the kernel f?

Filter Dimensions (3, 3)

input Dimensions (6, 6)

Number of parameters in the kernel = 3\*3 = 9

2. What is the output activation map when you apply the convolutional operation using the filter f on the input X without padding?

Output Activation Map

[[ 16 9 -4 -18]

[ 17 -5 -10 -12]

[ 11 -9 -17 2]

[ 9 -1 -15 16]]

3. What is the output when you apply a max-pooling operation on the output from the

previous question?

Output when you apply a max pooling

[[17 -4]

[11 16]]