Report

Burak Akgül

24.04.2021

1 Sanity Checks

Since we know that our dataset is balanced, we can do the following checks.

1.1 Loss

Printing the loss at the beginning is an option, as we should see that the losses are nearly equal to each other because we only just began the training. For the first batch, the first loss was 2.303. If we take -log(0.1) (because there are 10 classes and data is seperated to those classes equally), we would see the expected result should have been approximately 2.302. The loss is nearly the same, meaning our model initialized correctly.

1.2 Accuracy

We expect the accuracy to begin at nearly 10% as there are 10 classes and guessing will be random at the beginning.The accuracy begins at approximately $8\% - 10\%$ for the first batch, which is similar to our expectations and validates our point.

2 Separating Validation Set

20% of the whole data is seperated as validation set and 80% as training set.

3 Hyperparameter optimization

3.1 1-layer (0-hidden-layer) network

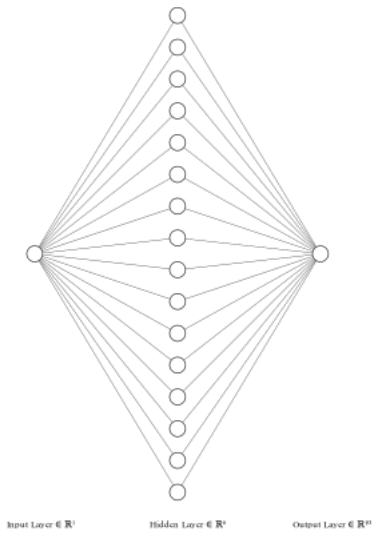
For the model architecture, 1 layer with 4800 input size and 10 output (class number) size is chosen. The learning rates are chosen to be the 0.0003, 0.00019, 0.00012, 0.000076, 0.000048 and 0.00003 (dividing by $\sqrt{10}/2$) and the model is trained for 50 epochs. The resulting table is below:

AF and HS	Learning Rate					
	0.0003	0.00019	0.00012	0.000076	0.000048	0.00003
-, -	0.3901	0.3874	0.3919	0.3754	0.3844	0.3887

Table 1: 1-layer network

3.2 2-layer (1-hidden-layer) network

The network architecture is:



Where the input size is 4800, and the hidden layer sizes are one of the followings: $\{1024, 1536, 2048\}$ and the activation functions are: $\{sigmoid, tanh, relu\}$ between the layers. The learning rates are chosen to be the 0.0003, 0.00019, 0.00012, 0.000076, 0.000048 and 0.00003 (dividing by $\sqrt{10}/2$). The combinations of these parameters makes up to 54 different experiments where the found accuracies are given in the below table (where S:sigmod, T: tanh, R:relu activation functions):

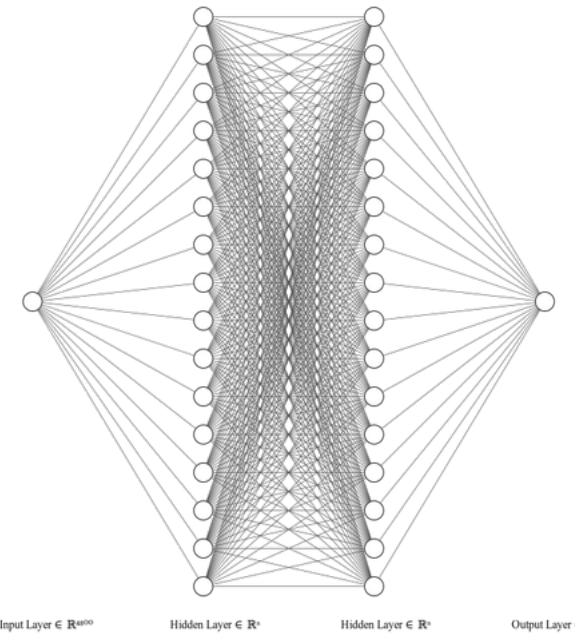
Layer Activations	Learning Rate					
	0.0003	0.00019	0.00012	0.000076	0.000048	0.00003
S, 1024	0.5699	0.5674	0.5601	0.5275	0.4677	0.4152
S, 1536	0.5718	0.5620	0.5516	0.5355	0.4778	0.4162
S, 2048	0.5733	0.5586	0.5518	0.5415	0.4758	0.4137
T, 1024	0.5511	0.5689	0.5701	0.5588	0.5360	0.5103
T, 1536	0.5526	0.5744	0.5538	0.5492	0.5470	0.5127
T, 2048	0.5555	0.5709	0.5596	0.5447	0.5329	0.5137
R, 1024	0.5570	0.5523	0.5641	0.5626	0.5558	0.5335
R, 1536	0.5706	0.5613	0.5648	0.5570	0.5535	0.5447
R, 2048	0.5649	0.5708	0.5621	0.5616	0.5536	0.5502

Table 2: 2-layer network

These experiments are tried for 50 epochs and the accuracies given are decided by the best validation accuracies of these 50 epochs.

3.3 3-layer (2-hidden-layer) network

The network architecture is:



Where the input size is 4800, and the hidden layer sizes are one of the followings: $\{1024, 1536, 2048\}$ and the activation functions are: $\{sigmoid, tanh, relu\}$ between the layers. The learning rates are chosen to be the 0.0003, 0.00019, 0.00012, 0.000076, 0.000048 and 0.00003 (dividing by $\sqrt{10}/2$). The combinations of these parameters makes up to 54 different experiments where the found accuracies are given in the below table (where S:sigmod, T: tanh, R:relu activation functions):

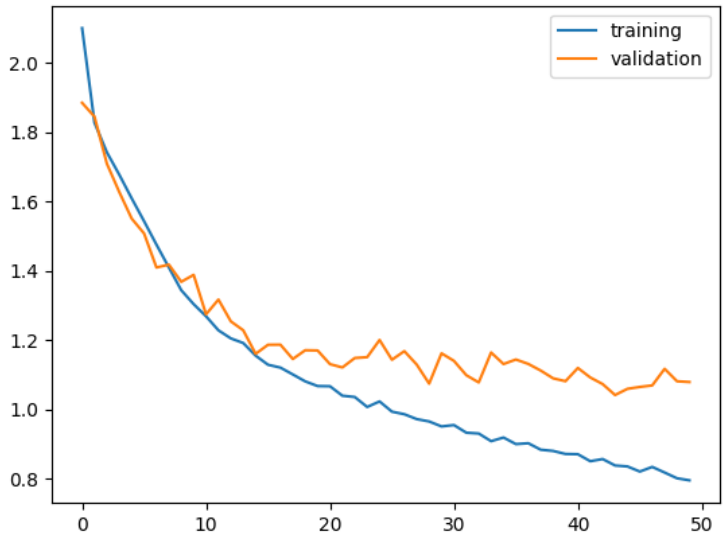
Layer Activations	Learning Rate					
	0.0003	0.00019	0.00012	0.000076	0.000048	0.00003
S, 1024	0.5485	0.5473	0.5555	0.5566	0.5202	0.4464
S, 1536	0.5531	0.5585	0.5526	0.5585	0.5423	0.4664
S, 2048	0.5458	0.5455	0.5555	0.5550	0.5470	0.4845
T, 1024	0.5533	0.5644	0.5759	0.5841	0.5834	0.5631
T, 1536	0.5543	0.5678	0.5706	0.5862	0.5879	0.5744
T, 2057	0.5275	0.5581	0.5792	0.5887	0.5925	0.58161
R, 1024	0.5887	0.5872	0.5756	0.5701	0.5723	0.5723
R, 1536	0.6020	0.5939	0.5743	0.5663	0.5625	0.5663
R, 2048	0.6048	0.5817	0.5691	0.5628	0.5578	0.5648

Table 3: 3-layer network

4 The best hyperparameter

4.1 Results

The hyperparameters for the best validation score 60.48% are Relu activation function with 2048 hidden layer size and 0.0003 learning rate. The test accuracy was 59.48%. The training and validation losses are given below:



4.2 Overfitting countermeasures

The countermeasure that I took against overfitting was saving the model when I have the best accuracy for my validation dataset. One can understand when overfitting occures by looking at losses and accuracies of training and validation, such that when overfit occurs, validation loss would be increasing and accuracy would be decreasing meanwhile training loss is decreasing and training accuracy would be increasing. From the plot, we can see that training accuracy drops but at approximately 20 epochs, validation loss would start to fluctuate and does not drop down much further. That is the point when the overfit starts to occur.

5 Comments