Report

Ahmet Dara Vefa

27.11.2020

1 Sanity Checks

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

1.1 Loss

What do you expect the loss to be? Calculate it. After the random initialization of the weights do not train the model and calculate the loss. Is it similar to what you expected?

PLEASE READ THE COMMENTS BEFORE READING THE REPORT.

I expect the loss to be around 4.6=-(ln(1/100)) since there are 100 classes.
I get around 4.6 loss on validation split without training the model(train.py lines 205-209).

1.2 Accuracy

After the random initialization of the weights do not train the model and calculate the accuracy on one of the splits and report it. Is it similar to what you expected?

I expect the accuracy to be around 1% since there are 100 classes and randomly selecting a class would give 1% accuracy.
Yes I get 1.05% accuracy without training the model.(train.py lines 205-209)

2 Hyperparameter optimization

For all tests, I used 5 epochs and didn't bother with implementing early stopping since epoch count is low. I also ran all tests with learning rate = [0.01,0.003,0.001, 0.0003, 0.0001, 0.00003] per model.

The training data set was taken as the first 80% of the given training images(first 8,000 images of the 10,000 images).
I used the same hidden layer size and activation function for both of the hidden layers of the 3 layer network.
The values at the tables are the minimum losses of each training result. You can see the training results at the results.txt file. The loss is calculated as sum(loss at batch)/(batch count) for more consistent results.

2.1 1-layer (0-hidden-layer) network

AF and HS	Learning Rate					
	0.01	0.003	0.001	0.0003	0.0001	0.00003
-, -	4.21	2.38	2.60	3.14	3.73	4.24

Table 1: 1-layer network

2.2 2-layer (1-hidden-layer) network

Layer Activations	Learning Rate					
	0.01	0.003	0.001	0.0003	0.0001	0.00003
S, 256	4.77	3.48	2.84	3.54	4.22	4.49
S, 512	4.92	3.64	2.66	3.28	4.05	4.45
S, 1024	5.23	3.67	2.53	3.04	3.84	4.39
T, 256	5.23	4.53	2.78	2.80	3.46	4.17
T, 512	5.79	4.53	4.62	2.54	3.19	3.98
T, 1024	7.11	4.61	2.58	2.34	2.95	3.75
R, 256	4.60	4.01	3.18	3.25	3.58	4.20
R, 512	4.60	3.56	2.90	2.98	3.34	4.03
R, 1024	4.60	3.77	2.75	2.82	3.11	3.82

Table 2: 2-layer network

2.3 3-layer (2-hidden-layer) network

Layer Activations	Learning Rate					
	0.01	0.003	0.001	0.0003	0.0001	0.00003
S, 256	4.60	4.36	3.10	3.74	4.48	4.58
S, 512	4.60	4.50	2.86	3.45	4.28	4.57
S, 1024	4.97	4.60	3.36	3.16	3.91	4.55
T, 256	5.23	4.80	2.87	2.72	3.34	4.08
T, 512	5.77	5.00	2.65	2.36	3.03	3.74
T, 1024	4.60	4.80	4.55	4.50	4.54	2.68
R, 256	4.60	3.26	2.67	3.12	3.46	4.18
R, 512	4.60	3.23	2.48	2.76	3.20	3.87
R, 1024	7.15	5.34	2.88	2.04	2.75	3.39

Table 3: 3-layer network

3 The best hyperparameter

3.1 Results

Give the hyperparameters of the network that achieved best in validation. Calculate its test accuracy and report it. Draw its training and validation loss graph (both loss should appear in the same graph so that we can compare).

The best network was three layer model with 0.0003 learning rate and tanh activation function.

After finding the best model, I trained it for 50 epochs and draw the results(Figure_50epoch.png). According to the graph, validation loss starts to increase at about 15 epochs. So I trained the model for 15 epochs and tested its accuracy on the validation split:
average evaluation loss = 3.7681180507906022, with 311 correct guesses. Accuracy = 15.55%
Testing this model on the submission gives 18% accuracy.
I also trained and tested models with different 15,20,25 epochs and noted them. You can see the results at validationAndTestResults.txt.
The highest test accuracy was achieved at 25 epochs with 21% test accuracy.(PS: I managed to get 24% accuracy at earlier versions of my models that was wrongly trained, hence my max accuracy at the leaderboard is 24% with a lucky model I guess.)

3.2 Overfitting countermeasures

What countermeasure did you take against overfitting? How may one understand when a network starts to overfit during training? You can use the train/validation loss graph in your explanation if you want.

I checked the aforementioned graph and found the lowest point of validation loss and trained my model based on that. (Although I got better validation and test accuracies with higher epochs.)

4 Comments

Since I am using windows my path variables are different then the ones shown in the tutorials.

I tried to get near 0.3 accuracy but I couldn't. I tried all the hyperparameters with 5 epochs and filled the tables accordingly. None of the correctly trained models with 2.4 loss achieved more than 21% accuracy with the submission. To check if I needed to train the models longer, I tried only ThreeLayerModel with 1024 hidden layer size with 15 epochs but the best model was the same with 5 epoch version. The values at the tables are at 5 epochs.
If you can, please let me know what my mistake was or please send me a model that works with 30% accuracy.