CENG499 - Introduction to Machine Learning
HW01 Report

Anıl Utku Ilgın
2260073

* Cells with best values are highlighted with red color.

Configuration	1 Layer L. Rate = 0.01 Epoch = 14	1 Layer L. Rate = 0.001 Epoch = 6	1 Layer L. Rate = 0.0001 Epoch = 20
	Val Loss: 2.74 Val Acc: 19.89%	Val Loss: 2.06 Val Acc: 28.70%	Val Loss: 2.01 Val Acc: 30.18%

Configuration	Activation Function = Tanh	Activation Function = Sigmoid	Activation Function = ReLU
2 Layer L.Rate = 0.01 H.Layer Neuron = 512	Val Loss: 2.66 Val Acc: 20.15%	Val Loss: 2.25 Val Acc: 27.32%	Val Loss: 2.78 Val Acc: 29.78%
2 Layer L.Rate = 0.01 H.Layer Neuron = 768	Val Loss: 3.09 Val Acc: 21.23%	Val Loss: 2.47 Val Acc: 25.36%	Val Loss: 3.34 Val Acc: 26.45%
2 Layer L.Rate = 0.01 H.Layer Neuron = 1024	Val Loss: 3.38 Val Acc: 21.95%	Val Loss: 3.08 Val Acc: 21.51%	Val Loss: 3.15 Val Acc: 28.54%
2 Layer L.Rate = 0.001 H.Layer Neuron = 512 Epoch = 12	Val Loss: 1.98 Val Acc: 36.94%	Val Loss: 1.78 Val Acc: 41.23%	Val Loss: 1.82 Val Acc: 42.61%
2 Layer	Val Loss: 1.92	Val Loss: 1.82	Val Loss: 1.80

L.Rate = 0.001 H.Layer Neuron = 768	Val Acc: 37.10%	Val Acc: 41.41%	Val Acc: 44.4%
2 Layer L.Rate = 0.001 H.Layer Neuron = 1024	Val Loss: 2.09 Val Acc: 35.56%	Val Loss: 1.85 Val Acc: 41.55%	Val Loss:1.82 Val Acc: 44.61%
2 Layer L.Rate = 0.0001 H.Layer Neuron = 512	Val Loss: 1.75 Val Acc: 40.01%	Val Loss: 1.87 Val Acc: 36.12%	**Val Loss: 1.55** **Val Acc: 46.70%** **Test Acc: 45.72%**
2 Layer L.Rate = 0.0001 H.Layer Neuron = 768	Val Loss: 1.72 Val Acc: 40.69%	Val Loss: 1.86 Val Acc: 36.19%	Val Loss: 1.57 Val Acc: 45.48%
2 Layer L.Rate = 0.0001 H.Layer Neuron = 1024	Val Loss: 1.72 Val Acc: 40.5%	Val Loss: 1.86 Val Acc: 36.8%	Val Loss: 1.59 Val Acc: 45.59%

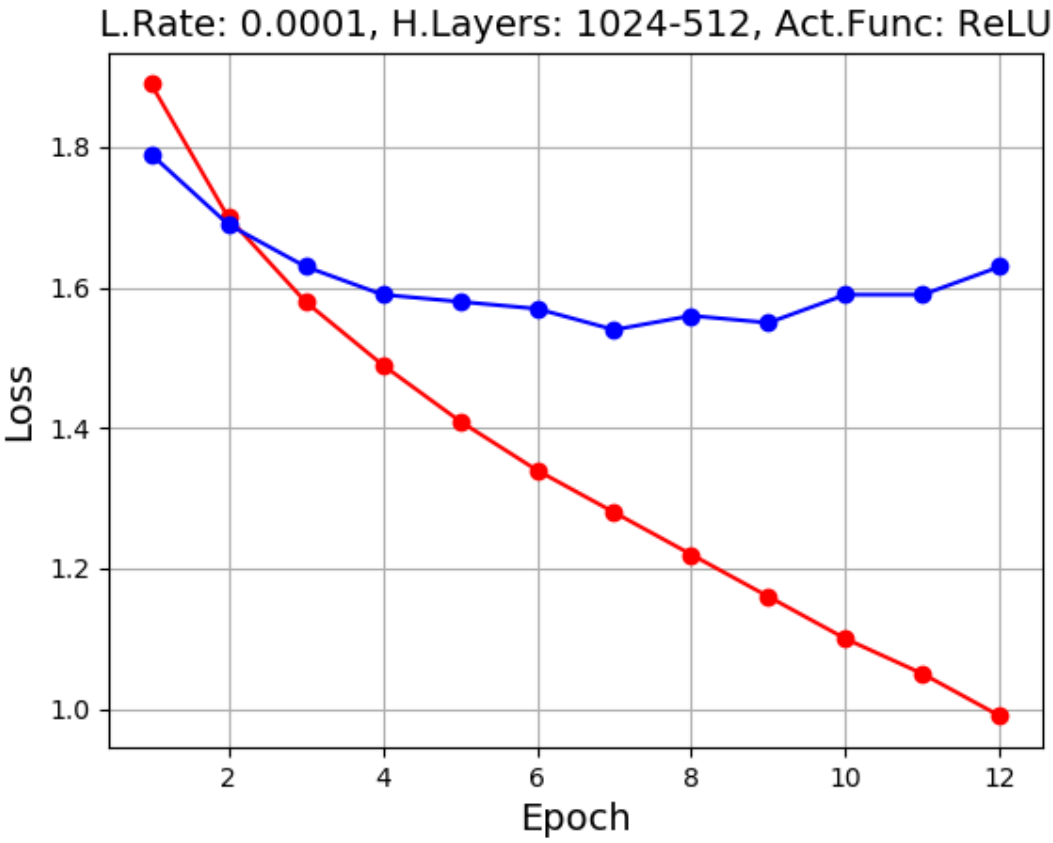
Configuration	Activation Function = Tanh	Activation Function = Sigmoid	Activation Function = ReLU
3 Layer L.Rate = 0.01 H.Layer 1 Neuron = 400 H.Layer 2 Neuron = 200	Val Loss: 2.28 Val Acc: 21.68%	Val Loss: 1.94 Val Acc: 31.95%	Val Loss: 2.15 Val Acc: 17.39%
3 Layer L.Rate = 0.01 H.Layer 1 Neuron = 512 H.Layer 2 Neuron = 256	Val Loss: 2.45 Val Acc: 20.81%	Val Loss: 2.01 Val Acc: 28.98%	Val Loss: 2.19 Val Acc: 15.70%
3 Layer L.Rate = 0.01 H.Layer 1 Neuron =	Val Loss: 2.55 Val Acc: 21.27%	Val Loss: 2.27 Val Acc: 23.40%	Val Loss: 2.30 Val Acc: 10%

1024 H.Layer 2 Neuron = 512			
3 Layer L.Rate = 0.001 H.Layer 1 Neuron = 400 H.Layer 2 Neuron = 200	Val Loss: 1.75 Val Acc: 40.87%	Val Loss: 1.80 Val Acc: 41.54%	Val Loss: 1.76 Val Acc: 44.46%
3 Layer L.Rate = 0.001 H.Layer 1 Neuron = 512 H.Layer 2 Neuron = 256	Val Loss: 1.79 Val Acc: 41.03%	Val Loss: 1.79 Val Acc: 41.92%	Val Loss: 1.73 Val Acc: 44.55%
3 Layer L.Rate = 0.001 H.Layer 1 Neuron = 1024 H.Layer 2 Neuron = 512	Val Loss: 1.77 Val Acc: 39.2%	Val Loss: 1.77 Val Acc: 40.81	Val Loss: 1.83 Val Acc: 43.42%
3 Layer L.Rate = 0.0001 H.Layer 1 Neuron = 400 H.Layer 2 Neuron = 200	Val Loss: 1.70 Val Acc: 41.27%	Val Loss: 1.76 Val Acc: 37.71%	Val Loss: 1.60 Val Acc: 45.20%
3 Layer L.Rate = 0.0001 H.Layer 1 Neuron = 512 H.Layer 2 Neuron = 256	Val Loss: 1.68 Val Acc: 42.14%	Val Loss:1.73 Val Acc: 38.55%	**Val Loss: 1.61** **Val Acc: 45.63%** **Test Acc: 44.78%**
3 Layer L.Rate = 0.0001 H.Layer 1 Neuron = 1024 H.Layer 2 Neuron = 512	Val Loss: 1.63 Val Acc: 44.32%	Val Loss: 1.69 Val Acc: 40.37%	**Val Loss: 1.63** **Val Acc: 46.39%** **Test Acc: 45.44%**

-Among the all three learning rates, the 0.0001 ones yield the best results in my configurations.
-Among all three activation functions (ReLU, Tanh, Sigmoid) the ReLU was yielding the best results in my configurations.
-Among all three layer configurations, the ones with 2 Hidden Layers yield the best results in my configurations.

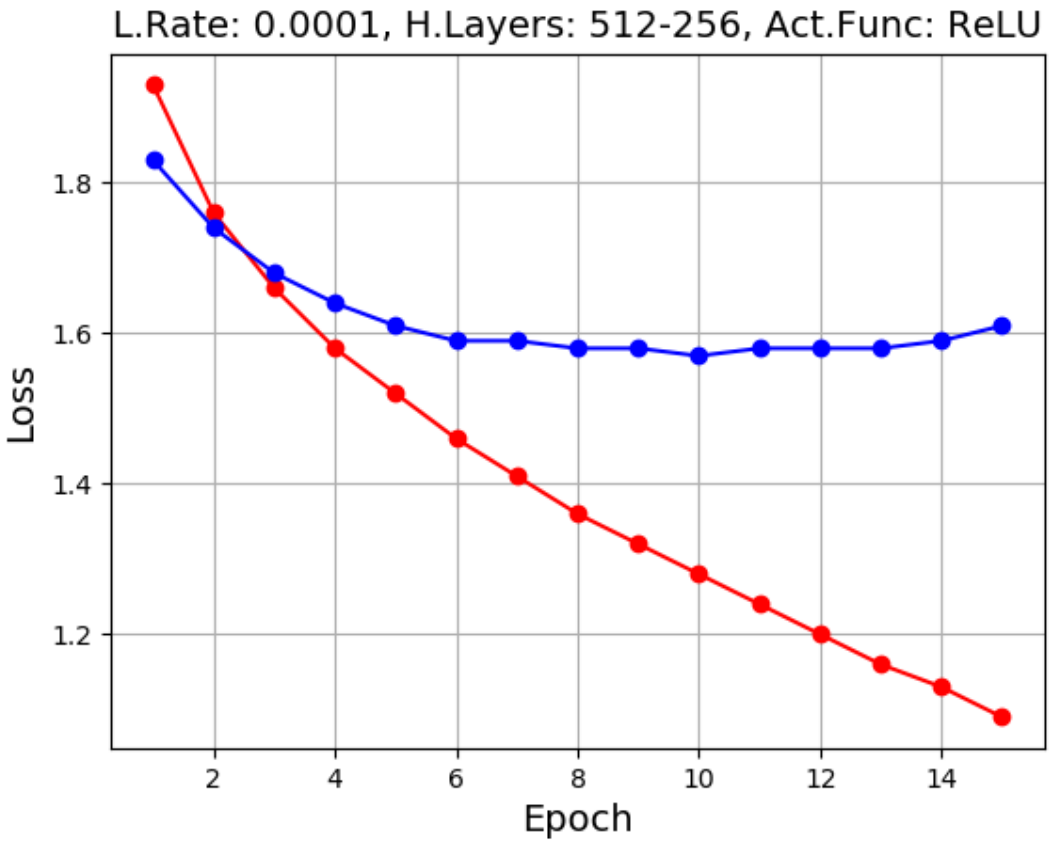
-Since among all these results, the same configuration was showing up as the best result, I decided to get the top three configuration for commenting.

The best one is the one with 2 Hidden Layers, ReLU function and L. Rate of 0.0001



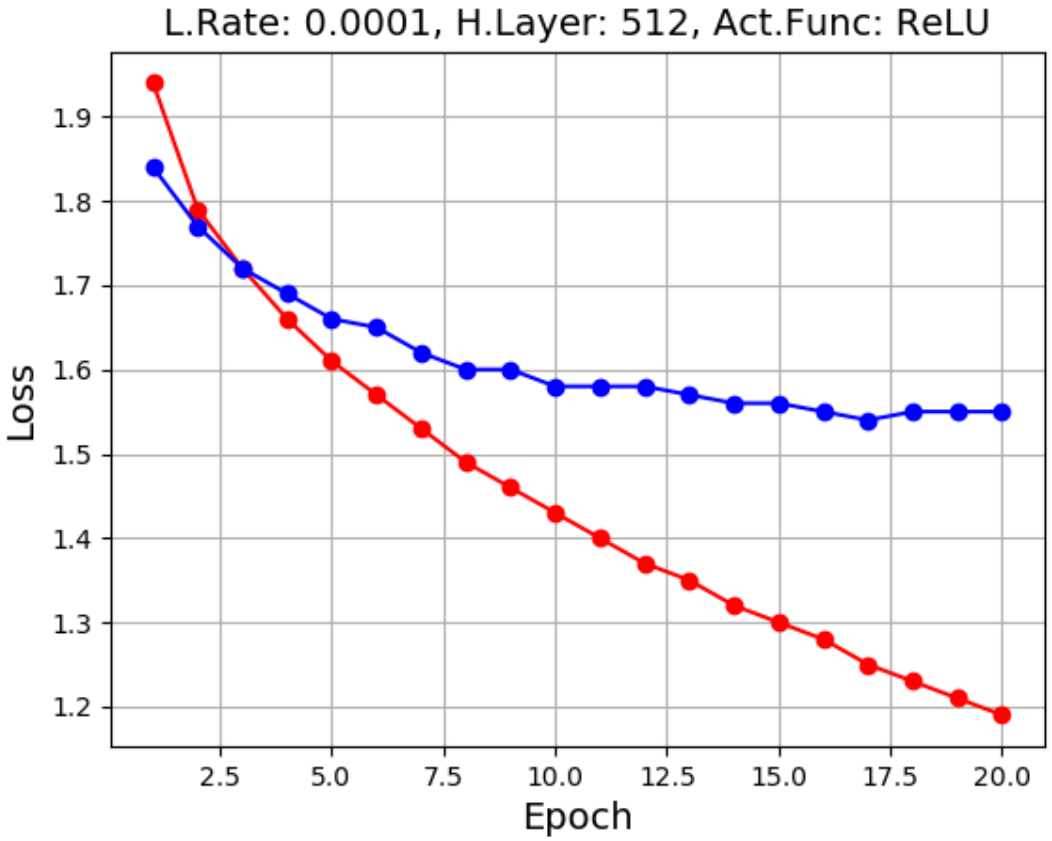
Comment

-The network started to overfit after the 2nd Epoch slightly, and the network was certainly overfitting after the 7th. The validation loss is diverging from the converging point, indicating a clear overfitting. The training loss seemed to continue to decrease, that's also strengthening the overfitting issue.



Comment

-The network started to overfit after the 10th Epoch, and was certain of it after the 15th. The validation loss is looking near a converging point. The training loss seemed to continue to decrease, and the difference between the validation and training loss is clearly indicating an overfitting. The network started to memorize the training data. Applying a dropout might provide better results to overcome the overfitting.



Comment

-This network didn't overfit, and may even be underfit since the training stopped after the 20th Epoch. The validation loss is looking near a converging point, however it might decrease even further by looking around the 16th epoch. The training loss seemed to continue to decrease, and the difference between the validation and training loss might indicate a potential overfitting, the network probably started to memorize the training data. The K-Fold method might yield better results in this case.

How did I overcome overfitting?

I decided to use early-stopping to overcome the overfitting issue.

How may one understand when a network starts to overfit during training?

An option is by observing the validation loss. If the validation loss is not decreasing, or more clearly converging to a point and then starting to increase again, the network is probably overfitting.

What method did you use to decide where to end training ?

I used early stopping with validation loss as the measurement. At every epoch, I checked if the current validation loss is the best until that time. If it's not, the counter is increased. If that counter hits a certain number (5, in my case) then the training would stop. The counter would reset if a new best value is encountered.

Is Accuracy a suitable metric to measure the performance of a network for this specific dataset?

-For this specific dataset, I think accuracy is a suitable metric since we are working on a N-classification problem. The accuracy of our networks on the test, validation and train sets individually should indicate a performance between the different networks. Since there are no groups, we can't use a confusion matrix; there would be no True Positive / Negative etc results, and the mean-square errors because of the problem's nature. We might use the Area Under Curve methods also, but they are also a method of accuracy, hence it supports the claim of accuracy's suitableness.

What are the advantages/disadvantages of using a small learning rate?

Advantages:

With small learning rates, we wouldn't miss the minimum loss values, since we won't be taking bigger steps in the valley.

Disadvantages:

The function would improve smaller with each step, may require more epochs, hence it might end up with overfitting.

What are the advantages/disadvantages of using a big learning rate?

Advantages:

Using a big learning rate can help us to avoid the local minimas, therefore can yield better results.

Disadvantages:

We may overlook the minimum values in the valley and hence it may not even find a suitable point.

What are the advantages/disadvantages of using a small batch size?

Advantages:

Using a smaller batch size will help us by making the training available if we have low memory and CPU/GPU power.

Disadvantages:

IT'll consume more time, and probably yield a worse performance in backpropagation.

What are the advantages/disadvantages of using a bigger batch size?

Advantages:

Using a bigger batch size will help us by making the training faster, and the backpropagation more smooth hence can result in a better performance of network.

Disadvantages:

It'll consume more memory and CPU/GPU power.