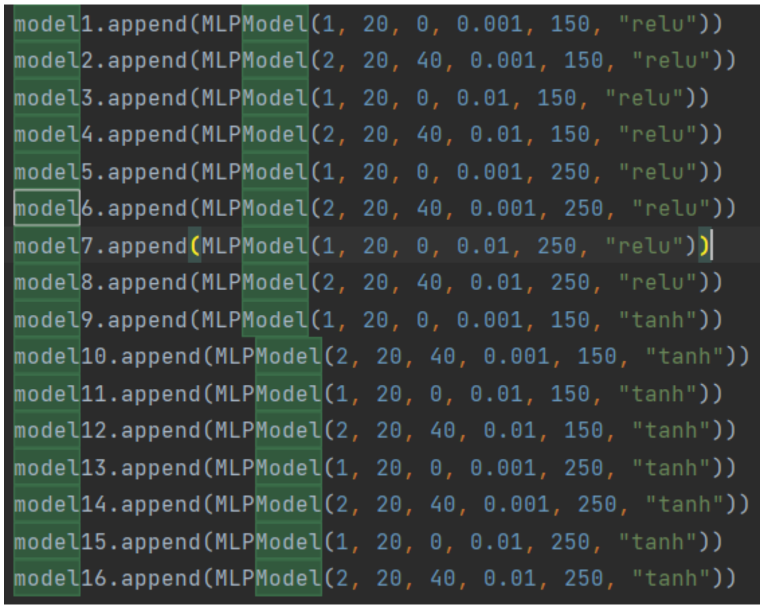
CENG499 Assignment1 Part3

Uygar Yaşar 2310613

November 2022

1 COMMENTS

1.1 First 16 model training



```
model1.append(MLPModel(1, 20, 0, 0.001, 150, "relu"))
model2.append(MLPModel(2, 20, 40, 0.001, 150, "relu"))
model3.append(MLPModel(1, 20, 0, 0.01, 150, "relu"))
model4.append(MLPModel(2, 20, 40, 0.01, 150, "relu"))
model5.append(MLPModel(1, 20, 0, 0.001, 250, "relu"))
model6.append(MLPModel(2, 20, 40, 0.001, 250, "relu"))
model7.append(MLPModel(1, 20, 0, 0.01, 250, "relu"))
model8.append(MLPModel(2, 20, 40, 0.01, 250, "relu"))
model9.append(MLPModel(1, 20, 0, 0.001, 150, "tanh"))
model10.append(MLPModel(2, 20, 40, 0.001, 150, "tanh"))
model11.append(MLPModel(1, 20, 0, 0.01, 150, "tanh"))
model12.append(MLPModel(2, 20, 40, 0.01, 150, "tanh"))
model13.append(MLPModel(1, 20, 0, 0.001, 250, "tanh"))
model14.append(MLPModel(2, 20, 40, 0.001, 250, "tanh"))
model15.append(MLPModel(1, 20, 0, 0.01, 250, "tanh"))
model16.append(MLPModel(2, 20, 40, 0.01, 250, "tanh"))
```

Figure 1: First 16 model training results

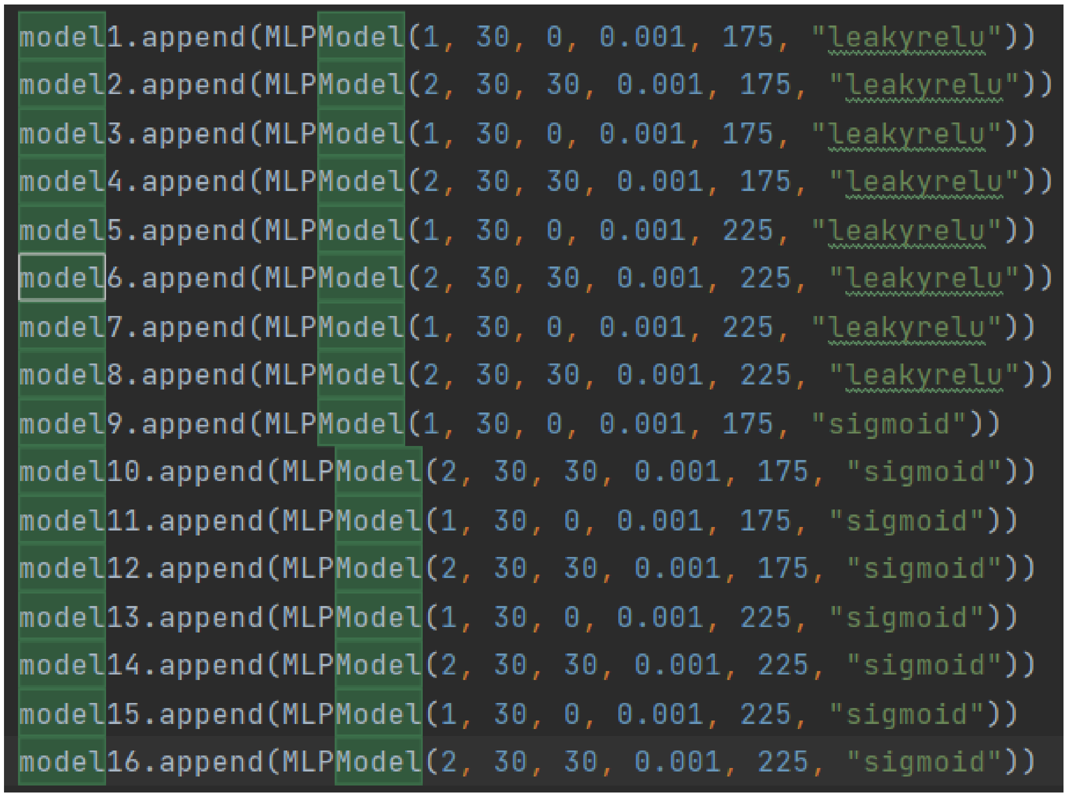
- The most remarkable difference happens because of the change of the learning rate. 0.001 is significantly better than 0.01.

- 150 epoch seems better than 250. However, more optimal epoch can be found and it can be changed according to other hyperparameters.

- One hidden layer gives better performance with every configuration.

- Relu gave better result in every case.

1.2 Second 16 model training



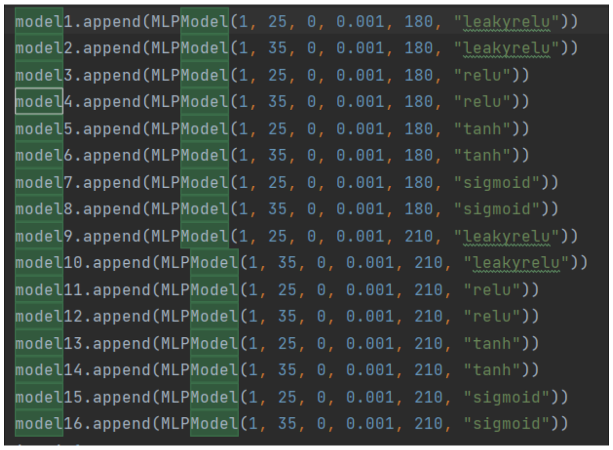
```
model1.append(MLPModel(1, 30, 0, 0.001, 175, "leakyrelu"))
model2.append(MLPModel(2, 30, 30, 0.001, 175, "leakyrelu"))
model3.append(MLPModel(1, 30, 0, 0.001, 175, "leakyrelu"))
model4.append(MLPModel(2, 30, 30, 0.001, 175, "leakyrelu"))
model5.append(MLPModel(1, 30, 0, 0.001, 225, "leakyrelu"))
model6.append(MLPModel(2, 30, 30, 0.001, 225, "leakyrelu"))
model7.append(MLPModel(1, 30, 0, 0.001, 225, "leakyrelu"))
model8.append(MLPModel(2, 30, 30, 0.001, 225, "leakyrelu"))
model9.append(MLPModel(1, 30, 0, 0.001, 175, "sigmoid"))
model10.append(MLPModel(2, 30, 30, 0.001, 175, "sigmoid"))
model11.append(MLPModel(1, 30, 0, 0.001, 175, "sigmoid"))
model12.append(MLPModel(2, 30, 30, 0.001, 175, "sigmoid"))
model13.append(MLPModel(1, 30, 0, 0.001, 225, "sigmoid"))
model14.append(MLPModel(2, 30, 30, 0.001, 225, "sigmoid"))
model15.append(MLPModel(1, 30, 0, 0.001, 225, "sigmoid"))
model16.append(MLPModel(2, 30, 30, 0.001, 225, "sigmoid"))
```

Figure 2: Second 16 model training results

- One hidden layer still seems better in any case but the difference is less for this time. Probably it is because of the neuron count. Moreover, standart deviation is a bit higher in the models with two hidden layers.

- There is no big difference between 175 and 225 epoch. (Generally, sigmoid function gives better results in 225 epochs but leakyrelu gives better with 175 epochs)

1.3 Third 16 model training



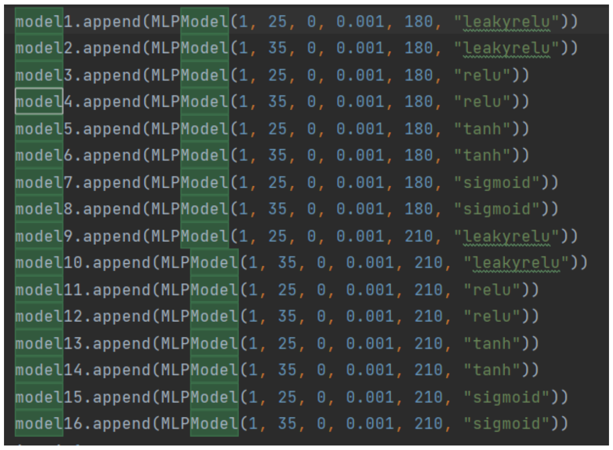
```
model1.append(MLPModel(1, 25, 0, 0.001, 180, "leakyrelu"))
model2.append(MLPModel(1, 35, 0, 0.001, 180, "leakyrelu"))
model3.append(MLPModel(1, 25, 0, 0.001, 180, "relu"))
model4.append(MLPModel(1, 35, 0, 0.001, 180, "relu"))
model5.append(MLPModel(1, 25, 0, 0.001, 180, "tanh"))
model6.append(MLPModel(1, 35, 0, 0.001, 180, "tanh"))
model7.append(MLPModel(1, 25, 0, 0.001, 180, "sigmoid"))
model8.append(MLPModel(1, 35, 0, 0.001, 180, "sigmoid"))
model9.append(MLPModel(1, 25, 0, 0.001, 210, "leakyrelu"))
model10.append(MLPModel(1, 35, 0, 0.001, 210, "leakyrelu"))
model11.append(MLPModel(1, 25, 0, 0.001, 210, "relu"))
model12.append(MLPModel(1, 35, 0, 0.001, 210, "relu"))
model13.append(MLPModel(1, 25, 0, 0.001, 210, "tanh"))
model14.append(MLPModel(1, 35, 0, 0.001, 210, "tanh"))
model15.append(MLPModel(1, 25, 0, 0.001, 210, "sigmoid"))
model16.append(MLPModel(1, 35, 0, 0.001, 210, "sigmoid"))
```

Figure 3: Third 16 model training results

- Sigmoid function works better with more nodes and more epochs.

- Epoch count does not have a huge effect on accuracy if it is in a certain range.

1.4 Fourth 16 model training



```
model1.append(MLPModel(1, 25, 0, 0.001, 180, "leakyrelu"))
model2.append(MLPModel(1, 35, 0, 0.001, 180, "leakyrelu"))
model3.append(MLPModel(1, 25, 0, 0.001, 180, "relu"))
model4.append(MLPModel(1, 35, 0, 0.001, 180, "relu"))
model5.append(MLPModel(1, 25, 0, 0.001, 180, "tanh"))
model6.append(MLPModel(1, 35, 0, 0.001, 180, "tanh"))
model7.append(MLPModel(1, 25, 0, 0.001, 180, "sigmoid"))
model8.append(MLPModel(1, 35, 0, 0.001, 180, "sigmoid"))
model9.append(MLPModel(1, 25, 0, 0.001, 210, "leakyrelu"))
model10.append(MLPModel(1, 35, 0, 0.001, 210, "leakyrelu"))
model11.append(MLPModel(1, 25, 0, 0.001, 210, "relu"))
model12.append(MLPModel(1, 35, 0, 0.001, 210, "relu"))
model13.append(MLPModel(1, 25, 0, 0.001, 210, "tanh"))
model14.append(MLPModel(1, 35, 0, 0.001, 210, "tanh"))
model15.append(MLPModel(1, 25, 0, 0.001, 210, "sigmoid"))
model16.append(MLPModel(1, 35, 0, 0.001, 210, "sigmoid"))
```

Figure 4: Fourth 16 model training results (to determine the configuration with the best accuracy)

2 LOG SUMMARIES

Following are the summaries of logs of each figure in the first part.

2.1 Logs of the First figure

1- Mean of test accuracies: 83.80599212646484 Confidence interval: 80.4800033569336 85.55999755859375
2- Mean of test accuracies: 79.68400573730469 Confidence interval: 73.91000366210938 85.56999969482422
3- Mean of test accuracies: 60.9890022277832 Confidence interval: 58.25 66.31999969482422
4- Mean of test accuracies: 58.84699630737305 Confidence interval: 52.75 63.38999938964844
5- Mean of test accuracies: 81.54499816894531 Confidence interval: 79.62999725341797 83.97000122070312
6- Mean of test accuracies: 77.36000061035156 Confidence interval: 70.62999725341797 80.69999694824219
7- Mean of test accuracies: 51.57899856567383 Confidence interval: 46.86000061035156 55.689998626708984
8- Mean of test accuracies: 52.2609977722168 Confidence interval: 49.72999954223633 54.2400016784668
9- Mean of test accuracies: 81.21599578857422 Confidence interval: 79.43000030517578 83.27999877929688
10- Mean of test accuracies: 78.79399108886719 Confidence interval: 74.58999633789062 82.11000061035156
11- Mean of test accuracies: 57.66899871826172 Confidence interval: 50.869998931884766 61.95000076293945
12- Mean of test accuracies: 53.27099609375 Confidence interval: 46.88999938964844 58.66999816894531
13- Mean of test accuracies: 80.12800598144531 Confidence interval: 75.13999938964844 83.62000274658203
14- Mean of test accuracies: 73.26200103759766 Confidence interval: 67.8499984741211 77.08000183105469
15- Mean of test accuracies: 48.944000244140625 Confidence interval: 46.279998779296875 51.15999984741211
16- Mean of test accuracies: 46.0469970703125 Confidence interval: 43.52000045776367 48.650001525878906

2.2 Logs of the Second figure

1- Mean of test accuracies: 82.71598815917969 Confidence interval: 80.70999908447266 84.47000122070312
2- Mean of test accuracies: 80.7750015258789 Confidence interval: 77.93000030517578 84.29000091552734
3- Mean of test accuracies: 83.03199768066406 Confidence interval: 81.52999877929688 84.81999969482422
4- Mean of test accuracies: 80.1240005493164 Confidence interval: 74.16999816894531 84.63999938964844
5- Mean of test accuracies: 80.56401062011719 Confidence interval: 77.62000274658203 82.91000366210938
6- Mean of test accuracies: 77.71000671386719 Confidence interval: 69.31999969482422 81.72000122070312
7- Mean of test accuracies: 80.87199401855469 Confidence interval: 78.68000030517578 83.55999755859375
8- Mean of test accuracies: 77.89200592041016 Confidence interval: 73.4000015258789 81.9000015258789
9- Mean of test accuracies: 81.6729965209961 Confidence interval: 78.91999816894531 83.25
10- Mean of test accuracies: 68.00999450683594 Confidence interval: 61.88999938964844 74.30000305175781
11- Mean of test accuracies: 81.82699584960938 Confidence interval: 80.68000030517578 83.16999816894531
12- Mean of test accuracies: 67.06900024414062 Confidence interval: 55.790000915527344 75.80000305175781
13- Mean of test accuracies: 82.81999969482422 Confidence interval: 81.33000183105469 83.62999725341797
14- Mean of test accuracies: 72.9310073852539 Confidence interval: 70.79000091552734 74.88999938964844
15- Mean of test accuracies: 83.3270034790039 Confidence interval: 82.19000244140625 84.29000091552734
16- Mean of test accuracies: 73.7459945678711 Confidence interval: 68.86000061035156 77.44000244140625

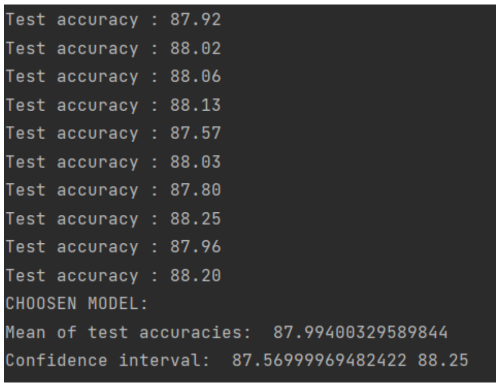
2.3 Logs of the Third figure

```
1-Mean of test accuracies: 82.8759994506836 Confidence interval: 81.12000274658203
84.94999694824219
2- Mean of test accuracies: 81.41500091552734 Confidence interval: 79.22000122070312
84.05000305175781
3- Mean of test accuracies: 82.46299743652344 Confidence interval: 79.47000122070312
84.9000015258789
4- Mean of test accuracies: 81.7669906616211 Confidence interval: 78.79000091552734
84.66000366210938
5- Mean of test accuracies: 82.22100067138672 Confidence interval: 78.05999755859375
83.94000244140625
6- Mean of test accuracies: 81.6659927368164 Confidence interval: 79.2300033569336
83.26000213623047
7- Mean of test accuracies: 81.25499725341797 Confidence interval: 78.3499984741211
82.70999908447266
8- Mean of test accuracies: 82.26699829101562 Confidence interval: 80.56999969482422
83.98999786376953
9- Mean of test accuracies: 82.19099426269531 Confidence interval: 79.23999786376953
83.62000274658203
10- Mean of test accuracies: 81.29199981689453 Confidence interval: 79.86000061035156
82.80000305175781
11- Mean of test accuracies: 81.94500732421875 Confidence interval: 78.23999786376953
85.06999969482422
12- Mean of test accuracies: 81.74500274658203 Confidence interval: 79.02999877929688
83.31999969482422
13- Mean of test accuracies: 80.80699920654297 Confidence interval: 70.62999725341797
84.5199966430664
14- Mean of test accuracies: 80.64100646972656 Confidence interval: 76.05999755859375
83.37000274658203
15- Mean of test accuracies: 80.57600402832031 Confidence interval: 78.05999755859375
82.68000030517578
16- Mean of test accuracies: 82.54500579833984 Confidence interval: 81.52999877929688
84.06999969482422
```

2.4 Logs of the Fourth figure

```
1-Mean of test accuracies: 83.06800079345703 Confidence interval: 81.37999725341797
84.48999786376953
2-Mean of test accuracies: 83.09100341796875 Confidence interval: 80.9000015258789
84.56999969482422
3-Mean of test accuracies: 82.35399627685547 Confidence interval: 79.30000305175781
84.2300033569336
4-Mean of test accuracies: 83.0669937133789 Confidence interval: 79.5999984741211
84.48999786376953
5-Mean of test accuracies: 81.11000061035156 Confidence interval: 76.98999786376953
84.55999755859375
6-Mean of test accuracies: 82.83499908447266 Confidence interval: 81.36000061035156
85.41000366210938
7-Mean of test accuracies: 81.99300384521484 Confidence interval: 80.55000305175781
83.01000213623047
8-Mean of test accuracies: 78.90800476074219 Confidence interval: 72.47000122070312
83.13999938964844
9-Mean of test accuracies: 82.47300720214844 Confidence interval: 79.91999816894531
83.79000091552734
10-Mean of test accuracies: 82.76699829101562 Confidence interval: 80.91000366210938
84.41999816894531
11-Mean of test accuracies: 83.62899780273438 Confidence interval: 82.3499984741211
84.79000091552734
12-Mean of test accuracies: 83.08699035644531 Confidence interval: 80.08000183105469
85.44999694824219
13-Mean of test accuracies: 82.31099700927734 Confidence interval: 80.22000122070312
84.44999694824219
14-Mean of test accuracies: 83.4020004272461 Confidence interval: 79.48999786376953
84.94999694824219
15-Mean of test accuracies: 82.55999755859375 Confidence interval: 81.6500015258789
83.5
16-Mean of test accuracies: 74.39399719238281 Confidence interval: 71.25 77.19000244140625
```

3 COMMENTS ON FINAL MODEL

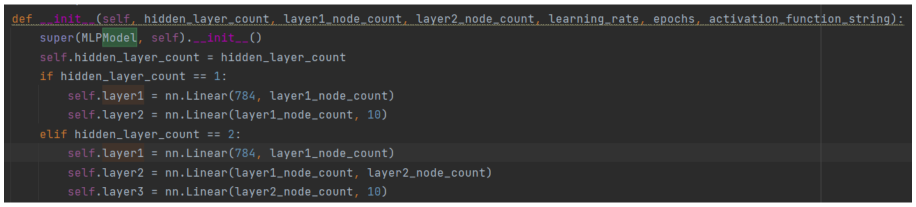


```
Test accuracy : 87.92
Test accuracy : 88.02
Test accuracy : 88.06
Test accuracy : 88.13
Test accuracy : 87.57
Test accuracy : 88.03
Test accuracy : 87.80
Test accuracy : 88.25
Test accuracy : 87.96
Test accuracy : 88.20
CHOOSEN MODEL:
Mean of test accuracies:  87.99400329589844
Confidence interval:  87.56999969482422 88.25
```

Figure 5: Model that gives highest performance trained with whole data

After merging both train data and validation data, obviously our data expands. So, model fed with larger data gives more accurate results. As it can be seen from first line of the first log and figure 5, the test accuracy becomes larger. (From 83.81 to 87.99). Moreover, confidence interval shrinks with larger data and it means that the results are more reliable.

4 IMPLEMENTATION



```python
def __init__(self, hidden_layer_count, layer1_node_count, layer2_node_count, learning_rate, epochs, activation_function_string):
    super(MLPModel, self).__init__()
    self.hidden_layer_count = hidden_layer_count
    if hidden_layer_count == 1:
        self.layer1 = nn.Linear(784, layer1_node_count)
        self.layer2 = nn.Linear(layer1_node_count, 10)
    elif hidden_layer_count == 2:
        self.layer1 = nn.Linear(784, layer1_node_count)
        self.layer2 = nn.Linear(layer1_node_count, layer2_node_count)
        self.layer3 = nn.Linear(layer2_node_count, 10)
```

Figure 6: Parameters of MLP class

In my implementation there is an MLP class with parameters in figure 6. So that I could change all the hyperparameters just by giving different parameters while creating an instance of that class. Then I created objects of this class. Since, training process will be 10 times and we must take the average test accuracy, I created arrays which includes copies of the same object. I separated training process into pieces with 16 different objects then compared their accurarcies for tuning. After interpreting these results then I did it with other objects with different parameters. I have some lists not to lose the average accuracy values. Then, I calculate confidence intervals. All my interpretations are in the first 3 parts.

5 ANSWERS TO QUESTIONS IN PDF FILE

1- Learning rate and epoch number, if learning rate and/or epoch number becomes too large then the validation accuracy started to decrease.

2- If validation accuracy decreased while train accuracy increasing, then the model starts to overfitting. Probably, test accuracy will also decrease.

3- If epoch number must be a relatively higher value, learning rate should be chosen small to prevent overfitting.

4- No, learning rate is a parameter and it needs to be tuning fine according to other hyperparameters in the configuration.

5- There is no such activation function that gives the best value all the time, if it were ever existed other activation functions will not be used. (For example, In my case sigmoid function gave better results with higher epoch numbers but relu had more accuracy for smaller number of epochs.)

6,7- Setting learning rate too low, model needs too many iterations and it causes long and costly learning processes, on the other hand it may give more accurate results. Setting it too high causes unstable learning process. For example, accuracies become 50 to 70 then 40 again within few epochs.

8- Using stochastic gradient descent learning can be computationally too costly for a large data set because it needs a loop through data in every step. So, it may not be a good idea.

9- The grayscale value is normalized because for example, sigmoid function inludes e^{-z} and it goes through zero when z becomes larger. So the sigmoid function may return exactly 1 or logarithmic functions may go through 1 against our will. So, we should feed those functions with relatively small values to get better results.