Introduction to Machine Learning

Necat Kılıçarslan - 2380624

November 2022

1 Part3 Report

In this report, I performed extensive parameter search over the MLP hyperparameters. These hyperparameters are number of hidden layers, number of neurons in these layers, learning rate, number of iterations and activation functions. In this report we analyze all mean configurations of 10 different modules. Also, we dramatically analyze the confidience interval of the 10 different modules. Lets first analyze the hyperparameters values.

- Number of hiddes layers parameters are 1 and 2.
- Number of neurons are 8 and 16.
- Learning rates are 0.001, 0.002 and 0.01
- Activation functions are nn.LeakyReLu and nn.PReLu
- Number of iterations are 250 and 500

These are the all hyperparameters which I used in this part. Then, lets share the model parameters of our models.

- Model1 $->$ Number of hidden layers: 1 , Number of neurons: 8 , Learning rate: 0.001 , Activation functions nn.LeakyRelu , Number of iterations: 250.
- Model2 $->$ Number of hidden layers: 1 , Number of neurons: 16 , Learning rate: 0.001 , Activation functions nn.LeakyRelu , Number of iterations: 250.
- Model3 $->$ Number of hidden layers: 1 , Number of neurons: 8 , Learning rate: 0.002 , Activation functions nn.LeakyRelu , Number of iterations: 250.
- Model4 $->$ Number of hidden layers: 1 , Number of neurons: 8 , Learning rate: 0.001 , Activation functions nn.PReLu , Number of iterations: 250.

- Model5 $->$ Number of hidden layers: 1 , Number of neurons: 8 , Learning rate: 0.001 , Activation functions nn.LeakyRelu , Number of iterations: 500.
- Model6 $->$ Number of hidden layers: 2 , Number of neurons: 8 , Learning rate: 0.001 , Activation functions nn.LeakyRelu , Number of iterations: 250.
- Model7 $->$ Number of hidden layers: 2 , Number of neurons: 16 , Learning rate: 0.001 , Activation functions nn.LeakyRelu , Number of iterations: 250.
- Model8 $->$ Number of hidden layers: 2 , Number of neurons: 16 , Learning rate: 0.002 , Activation functions nn.LeakyRelu , Number of iterations: 250.
- Model9 $->$ Number of hidden layers: 2 , Number of neurons: 16 , Learning rate: 0.002 , Activation functions nn.PReLu , Number of iterations: 500.
- Model10 $->$ Number of hidden layers: 2 , Number of neurons: 16 , Learning rate: 0.01 , Activation functions nn.PReLu , Number of iterations: 500.

After the definition of the model lets share the each mean of the all models.

Model No	Mean Prediction
Model 1	53.483249
Model 2	58.408447
Model 3	43.400500
Model 4	59.325710
Model 5	47.028035
Model 6	51.975606
Model 7	54.542525
Model 8	44.149048
Model 9	39.551609
Model 10	34.871297

Then it is time to analyze all standard deviation of the our models one by one.

Model No	Standard Deviation
Model 1	9.266825
Model 2	10.734680
Model 3	12.899836
Model 4	9.087734
Model 5	15.268242
Model 6	10.100408
Model 7	10.823369
Model 8	10.876480
Model 9	9.815636
Model 10	6.851641

Then now we have all the values in order to calculate confidence interval of the our models. Lets analyze confidence range of the all our models.

Model No	Confidence Interval
Model 1	47.739598 - 59.226901
Model 2	51.755008- 65.061887
Model 3	35.405080 - 51.395921
Model 4	53.693061 - 64.958360
Model 5	37.564658 - 56.491412
Model 6	45.715294 - 58.235919
Model 7	47.834115 - 61.250935
Model 8	37.407719 - 50.890376
Model 9	33.467800 - 45.635418
Model 10	30.624595 - 39.117998

Now, we can easily see that model4 has the maximum mean among these models. If we retrain this model with new dataset, we get the following results:

- Mean Value: 80.9010920
- Standard Deviation: 2.757431342
- Confidence interval: 79.1920142 - 82.6101697

Now, it is time to answer the report questions.

- In order to prevent overfitting, I do not increase too much the number of iterations. When I increased too much the epoch, then I realized that the model memorize the training dataset. In order to prevent this situation, we have to find a convergence upper limit value.
- If the accuracy rate decreases while iteration number increases, we can say that a model starts to overfit
- If the training is stopped before the optimal time, the model will not have had time to learn the most important features from the training set and therefore provide poorly fitted solutions for both the training and the test sets.On the other hand, if the training is stopped after the optimal epoch, high performance will be achieved in the training set, but when looking at the test results, the model would not generalize correctly, as it has over-adjusted to the training set. There is an easy and widely used method to avoid these two issues. It is known as early stopping
- Actually we can't find the best learning rate immediately, but if we experiment a lot and try a lot, we can get a near-optimal learning rate.
- Since there are many almost unlimited activation functions, it will be impossible to understand which one gives the best results without trying all of them, but the activation function that gives good results can be found thanks to our experiments.
- A smaller learning rate may allow the model to learn a more optimal or even globally optimal set of weights but may take significantly longer to train, learning rate that is too small can cause the process to get stuck.
- A learning rate that is too large can cause the model to converge too quickly to a suboptimal solution, in the other hand a large learning rate allows the model to learn faster, at the cost of arriving on a sub-optimal final set of weights
- There are two ways in which gradient descent may be inefficient. These two are too many gradient descent updates are required, each gradient descent step is too expensive.
- The pixel values can range from 0 to 256. Each number represents a color code. When using the image as it is and passing through a Deep Neural Network, the computation of high numeric values may become more complex. To reduce this we can normalize the values to range from 0 to 1. In this way, the numbers will be small and the computation becomes easier and faster. As the pixel values range from 0 to 256, apart from 0 the range is 255. So dividing all the values by 255 will convert it to range from 0 to 1.