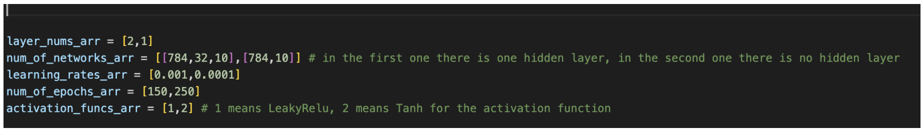
Introduction to Machine Learning Homework 1 Part 3

Nuray Akar 2380004

November 2022

1 Results



```
layer_nums_arr = [2,1]
num_of_networks_arr = [[784,32,10],[784,10]] # in the first one there is one hidden layer, in the second one there is no hidden layer
learning_rates_arr = [0.001,0.0001]
num_of_epochs_arr = [150,250]
activation_funcs_arr = [1,2] # 1 means LeakyRelu, 2 means Tanh for the activation function
```

As you can see from the figure above, I have choose these parameters and tried all possible configurations with them.
The best accuracy was the one below:
no hidden layer (784, 10) - learning rate: 0.001 - epoch number: 150 - activation function: LeakyRelu
trial 1:
Test - Loss 0.65
trial 2:
Test - Loss 0.65
trial 3:
Test - Loss 0.65
trial 4:
Test - Loss 0.65
trial 5:
Test - Loss 0.66
trial 6:
Test - Loss 0.67
trial 7:
Test - Loss 0.64
trial 8:
Test - Loss 0.66
trial 9:
Test - Loss 0.65
trial 10:
Test - Loss 0.67
Confidence Interval: (0.8478,0.8521)

After I re-train the model with the same configuration the result was:

```
trial 1:
Test - Loss 0.66
trial 2:
Test - Loss 0.66
trial 3:
Test - Loss 0.64
trial 4:
Test - Loss 0.67
trial 5:
Test - Loss 0.64
trial 6:
Test - Loss 0.65
trial 7:
Test - Loss 0.67
trial 8:
Test - Loss 0.66
trial 9:
Test - Loss 0.65
trial 10:
Test - Loss 0.66
Confidence Interval: (0.8469,0.8525)
```

2 Questions

1. What type of measure or measures have you considered to prevent overfitting?
   Initially, avoiding complex layers and high number of neurons prevents overfitting. Additionally, we should not choose a very high number for iteration.

2. How could one understand that a model being trained starts to overfit?
   While training looking validation loss would help us to understand if there is overfitting. If validation loss starts to increase, there may be overfitting. However, it is not always the case. Since there will be some local minima, we should try to find the global minimum point of the validation loss.

3. Could we get rid of the search over the number of iterations (epochs) hyperparameter by setting it to a relatively high value and doing some additional work? What may this additional work be? (Hint: You can think of this question together with the first one.)
   If we choose high value for the number of iterations, we should check the overall validation loss and find the global minimum. The proper number of iteration would be the value of that point.

4. Is there a "best" learning rate value that outperforms the other tested learning values in all hyperparameter configurations? (e.g it may always produce the smallest loss value and highest accuracy score among all of the tested hyperparameter configurations.). Please consider it separately for each task.
   I have used 0.001 and 0.0001 as learning rate. When I have used 0.001 as learning rate, the accuray results was better.

5. Is there a "best" activation function that outperforms the other tested activation functions in all hyperparameter configurations? (e.g it may always produce the smallest loss value and highest accuracy score among all of the tested hyperparameter configurations.). Please consider it separately for each task.
   I have used tanh and leakyrelu for activation function but there is not best in general. Sometimes tanh gives better results, sometimes otherwise.

6. What are the advantages and disadvantages of using a small learning rate?
   Advantages:

   (a) Possessing an extremely low probability of failing to reach the loss value's minimal point.

   Disadvantages:

   (a) Taking a lot of epochs to get to the least validation loss value point.

   (b) Having a greater propensity to encounter local minima rather than the global lowest point.

7. What are the advantages and disadvantages of using a big learning rate?
   Advantages:

   (a) Requiring fewer epochs to attain the validation loss value's minimal point.

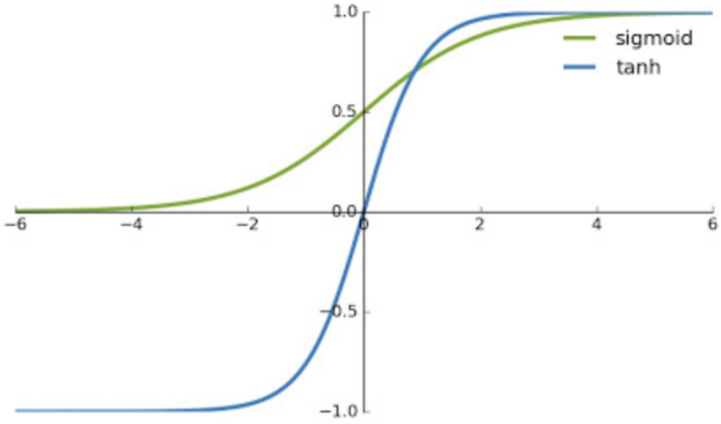
   (b) Being less likely to discover the global minimum than to fall into local minima.

   Disadvantages:

   (a) Having a greater chance of missing the minimum point of validation loss.

8. Is it a good idea to use stochastic gradient descent learning with a very large dataset? What kind of problem or problems do you think could emerge?
   Although SGD is substantially quicker than original gradient descent, its convergence route is noisier. This is due to the fact that it only approximatively calculates the gradient at each step. As a result, the cost fluctuates considerably. However, it is still a far superior option.

9. In the given source code, the instance features are divided by 255 (Please recall that in a gray scale-image pixel values range between 0 and 255). Why may such an operation be necessary? What would happen if we did not perform this operation? (Hint: These values are indirectly fed into the activation functions (e.g sigmoid, tanh) of the neuron units. What happens to the gradient values when these functions are fed with large values?)



As one can see from the graph above, after some values sigmoid and tanh converges to 1. Therefore, if we gave them values between 0 and 1, we will get different outputs for different inputs. Otherwise, different inputs will gave same outputs.