# ***Assignment 1***

# Neural Network basics, Multi-layer Perceptron, Gradient descent

Hui Xue

[hui.xue@nih.gov](mailto:hui.xue@nih.gov)

1. **Multi-choice and short-answer questions**
2. Given a four layer network, the first hidden layer has 3 neurons, the next two hidden layers each has 10 neurons, output layer has 1 neuron, input image has the dimension of 10x10x3, what is the total number of parameters?
3. 903+40+110+11
4. 903+40+40+11
5. 303+40+40+11
6. 303+40+110+11
7. Following steps are performed in the gradient descent, given a sample (x, y), please pick the correct order of conduction:
8. Backprop from loss to all parameters
9. Perform the gradient update step
10. Forward pass the network to compute network outputs
11. Compute loss by taking in model outputs and y
12. 1,2,3,4
13. 3,4,1,2
14. 1,3,2,4
15. 3,4,2,1
16. Deep learning introduces new components on top of “shallow” models. Which of the following descriptions are correct?
17. Deep learning is special because it introduced the non-linear activation functions.
18. Deep learning does not rely on feature engineering, but allows models to learn what is important from the data.
19. Deep learning adds more layers to compose a deep network.
20. Deep learning can handle high dimensional data.
21. Comparing two common activation functions: sigmoid and ReLU, which of the following descriptions are correct?
    * 1. Sigmoid is a better activation function, because it was published much earlier.
      2. Gradient of sigmoid function can be close to zero, causing difficulties to train the model. ReLU mitigates this problem.
      3. ReLU is a better activation function, because it is cheap to compute.
      4. You cannot mix two activation functions in one network.
22. Which of the following descriptions are correct, involving gradient descent (GD), stochastic gradient descent (SGD) and mini-batch gradient descent (MB-SGD):
    * 1. By comparing the smoothness of loss decaying curve, GD > SGD > MB-SGD
      2. By comparing the frequency of parameter updates, MB-SGD > SGD > GD
      3. By comparing the frequency of parameter updates, SGD > MB-SGD > GD
      4. By comparing the smoothness of loss decaying curve, GD > MB-SGD > SGD
23. If you increase the batch size, the learning rate should be increased as well. Please explain the reason.
24. Use your own language, explain what is bayes error, what is bias and variance?
25. Why do we need a hold-out test datasets? Can we use a part of training datasets for testing purpose, e.g. if you only have 1000 sample, can you train on N=1000 samples and test on randomly picked N=100 samples?
26. Given a symmetric matrix such that and the input. We have a loss function **.**

What is the derivative and ?

Let the sigmoid function be , change the loss function to be **,** compute the derivative.

1. You are designing a multi-layer perception network for an image classification problem, analyze your model and answer questions:
2. Suppose your MLP has 3 hidden layers with 5, 10 and 10 neurons each. For a **binary classification problem to detect whether there are human in the image**, what is the number of neuron at output layer?
3. Draw a network graph for this MLP, as we saw in the lecture.
4. Suppose the input image has dimension 32x32x1. The weight matrix and bias vector for the first hidden layer is and . What are the dimension of and?
5. For other layers (two hidden layers and output layer), what are the dimensions of their weight matrixes and bias vectors? (**,**
6. What are the total number of parameters?
7. After you developed this model, **the requirement is changed to classify whether there are man, woman or children in the image**. How will you change the design of your network?
8. After you modified the network design for the new requirement, what are the dimensions of all weights and biases? What are the total number of parameters?
9. In the lecture, we learned the cross-entropy loss for the binary and multi-class classification cases.

The sample for training is . Here is the input image andis the output of network**.** For the multi-class classification, we takes the convention thatis the logit**.** In the binary classification case, a sigmoid is applied afterwards In the multi-class case, a softmax is applied.

Please answer following questions about these loss functions:

1. For a batch size B=10, suppose you are training a MLP network. What is the dimension of and, for binary and multi-class cases?
2. For a batch of samples, write out the empirical loss for BCE and CE cases.
3. Both BCE\_loss and CE\_loss are derived from the cross-entropy:

Please proof that the cross-entropy can be decomposed as the sum of entropy of distribution p and KL-divergence .

Please proof

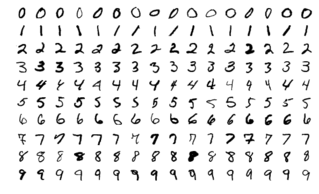
1. Let the empirical loss for a batch of samples be . Derive the derivatives of loss function to the logits , write out its dimensionality clearly. (Note is the output after applying sigmoid or softmax on logits. You will need to take derivatives through the sigmoid or softmax function.)
2. Typically, for a K class problem, the vector is the one-hot encoding to indicate the correct class. It could be useful to consider a different situation where has more than one non-zeros.

Design a loss function for this situation and write out the empirical loss for a mini-batch.

1. Coding problem

You will implement a MLP network to classify the MNIST (Modified [National Institute of Standards and Technology](https://en.wikipedia.org/wiki/National_Institute_of_Standards_and_Technology) database) dataset. This dataset is widely used to train and test different machine learning algorithms. You can learn more about MNIST at its wiki page: https://en.wikipedia.org/wiki/MNIST\_database

This dataset included handwritten digits from 0 to 9, with 50,000 images for training and 10,000 for testing. The dataset can be downloaded from <http://yann.lecun.com/exdb/mnist/>. Every image is a 28x28x1 grayscale picture, containing one of the 10 digits:



1. As a multi-class classification problem, given the size of a mini-batch is B, what is the output size of logits of the MLP? After passing the logits through the softmax, what is the size of the probabilities ? Clearly write out the final equation of empirical CE loss. Also, write out the derivative of empirical CE loss to the logits (network outputs before softmax). The answer of the previous question 4d may be useful here.
2. Following the instructions at the course website (https://deeplearningcrashcourse.org/nhlbi2021/) to set up your python development environment. You can use either VSCode(https://code.visualstudio.com/) or PyCharm(https://www.jetbrains.com/pycharm/) for development and debugging.

After setting up the environment, start your python environment and read through the problem py file (*a1\_mlp\_3\_layers.py*). Try to open this python file with VSCode and set break points to step through the code. Please check this video for instructions (<https://www.youtube.com/watch?v=W--_EOzdTHk>). You can also find other tutorials online.

1. The python file *a1\_mlp\_3\_layers.py* contains a 3 layer MLP model, unfinished. Please implement the empirical CE loss function, the forward and backward pass and the gradient descent step. The weights should be initialized from the standard norm distribution and biases are initialized as 0. The training loop has been implemented, including random shuffling the images and performing gradient descent. A portion of the training data was split out as the validation set, to estimate the model performance. After every epoch, the accuracy and loss values are saved.

Please finish the coding and train the model with default parameters (Batch Size, learning rate, number of epochs). **Submit the plots of the accuracy and loss vs. epoch for both training and validation sets.**

One way to debug your model is to let it fit on a single mini-batch with small batch size. Model should be able to fit the data extremely well, getting 100% accuracy and very low loss. Remember to turn off regularization when doing this test. Since the batch size is now small, you need to reduce the learning rate, e.g.

*python3 a1-mlp-3-layers.py --learning\_rate 0.01 --one\_batch\_training True --batch\_size 10 --num\_epochs 1000*

1. Next step is to add L2 regularization to the loss function. Note the L2 norm is computed on the weights only, not the biases. First, write out the empirical loss with L2 regularization as the format of “CE\_loss+. How will you change the gradient computation for weights? Implement the backprog function with regularization and retrain the model. **Submit the plots of the accuracy and loss curves vs. epoch for both training and validation sets**. What do you observe after adding regularization?
2. The python file *a1\_mlp\_N\_layers.py* contains a MLP model with any number of hidden layers, unfinished. Please implement the forward and backward pass and the gradient descent step, with the L2 regularization. Please finish the coding and train the model with default parameters (Batch Size, learning rate, number of epochs). Submit the plots of the accuracy **and loss vs. epoch for both training and validation sets. Compare to the accuracy you got on the 3-layer MLP, what are your findings?**

Now you can change the model architectures to try different number of layers or number of hidden units. Can you get higher accuracy? Explain what you tried.

1. A critical aspect of training models is the reproducibility in terms of model performance. A common tool for python regression test is the pytest package (<https://docs.pytest.org/en/6.2.x/contents.html>). Learn to use pytest to write test cases for your model. A good tutorial can be found at <https://www.youtube.com/watch?v=byaxg00Gf9I>. Finish the test function in the test\_*a1\_mlp\_3\_layers.py and run the test with `pytest -s src`. Submit the output of your test run.*