BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI

HYDERABAD CAMPUS

Second Semester 2020-21

BITS F464 - Machine Learning

Assignment - 2B

Group Members

Danish Mohammad 2018A7PS0103H

Mridul Kumar Rai 2018AAPS0359H

Ketan Goyal 2018A8PS0900H

Artificial Neural Networks

Problem Statement

1) In this assignment, you will have to implement a simple artificial neural network with at most two hidden layers from scratch that can perform multi-class (1-out-of-K) classification. Choose appropriate non-linear activation (such as sigmoid, tanh, ReLU, LeakyReLU, etc) functions for the hidden and output layers and try to achieve as good accuracy as possible.
2) Use Stochastic Gradient Descent or mini-batch Gradient Descent (with appropriate batch size) to train the model. As usual, you can make a random 70:30 split on the given data and use it for training and testing respectively. Plot the loss and accuracy of your model (advisedly in two separate plots, i.e. one for accuracy and one for loss) after every 50 iterations to visualize the training better.
3) Try to vectorize your code as much as possible to make your computations faster and efficient. Do not hard code any parts of the implementation unless it is absolutely necessary.

What needs to be documented

1) A very brief description of your model and its implementation.

To predict labels for a series of attributes, the model employs an Artificial Neural Network. This model is used to classify from multiple classes that are not linearly separable.

The dataset is read, shuffled, and divided into training and testing split of a 70:30 ratio. The `oneHotEncoding()` is used to determine one-hot encodings for the y-labels. The data is standardised to make mean 0 and standard deviation 1. The biases are set to 0 and the weights are initialised from a regular normal distribution using `parametersSetting()`. `feedforward()` is used to get all node values for the network and perform forward propagation. `grad_fn()` is used to measure the gradients for the parameters using the nodes

and training results. Finally, using the gradients (*grad_params*) and the learning rate (*eta*), backpropagation is used to update the parameters. The iteration's loss and accuracy are measured using the expected and training labels. The estimated preparation and testing labels are collected and their accuracies are measured at the end of gradient descent using the optimal weights and biases. After that, plots of losses and accuracy vs. epochs are made. The whole procedure is replicated for a two-layered network.

2) **A brief description of your chosen hyperparameters for the model such as number of hidden layers, number of units per layer, activation functions for each layer and learning rate.**

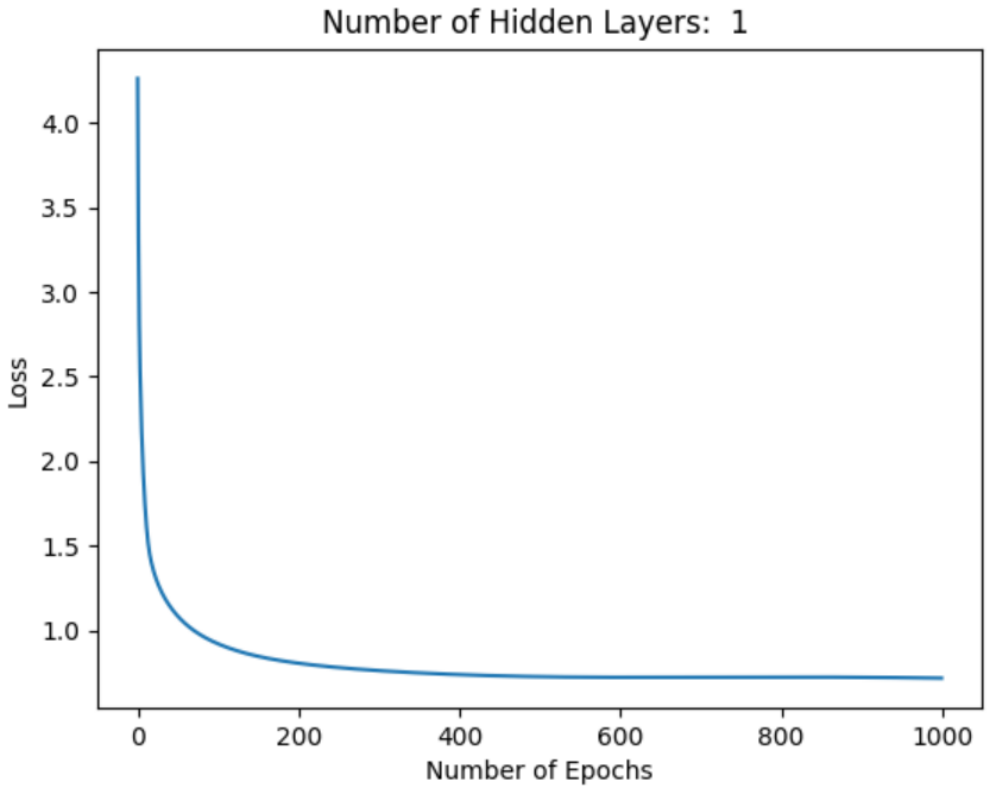
Learning rate(eta) = 0.5
Iterations= 1000
Number of hidden layers = 1 and 2
No. of inputs in the input layer= 6
No. of inputs in the 1st hidden layer = 20
No. of inputs in the 2nd hidden layer 2 = 20
Units in the output layer=10
Activation function for the 1st hidden layer= Sigmoid
Activation function for the 2nd hidden layer= Sigmoid
Activation function for the output layer= Softmax

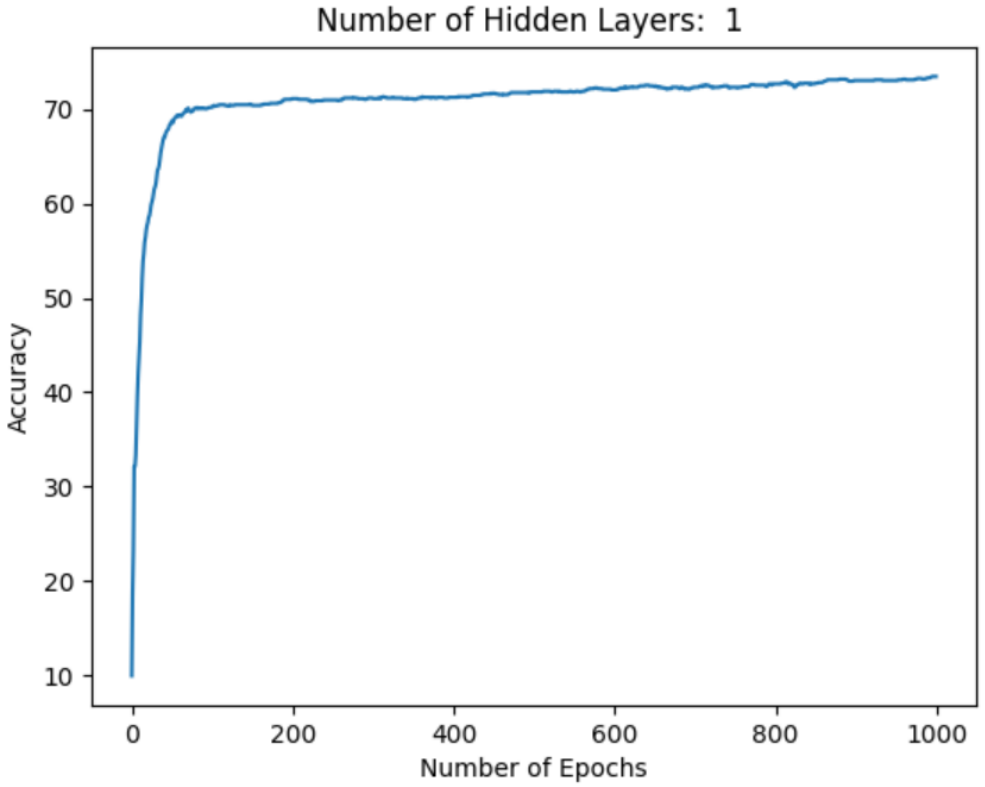
3) **The final train and test metrics (loss and accuracy) achieved by your model for ANN with one hidden layer and two hidden layers.**

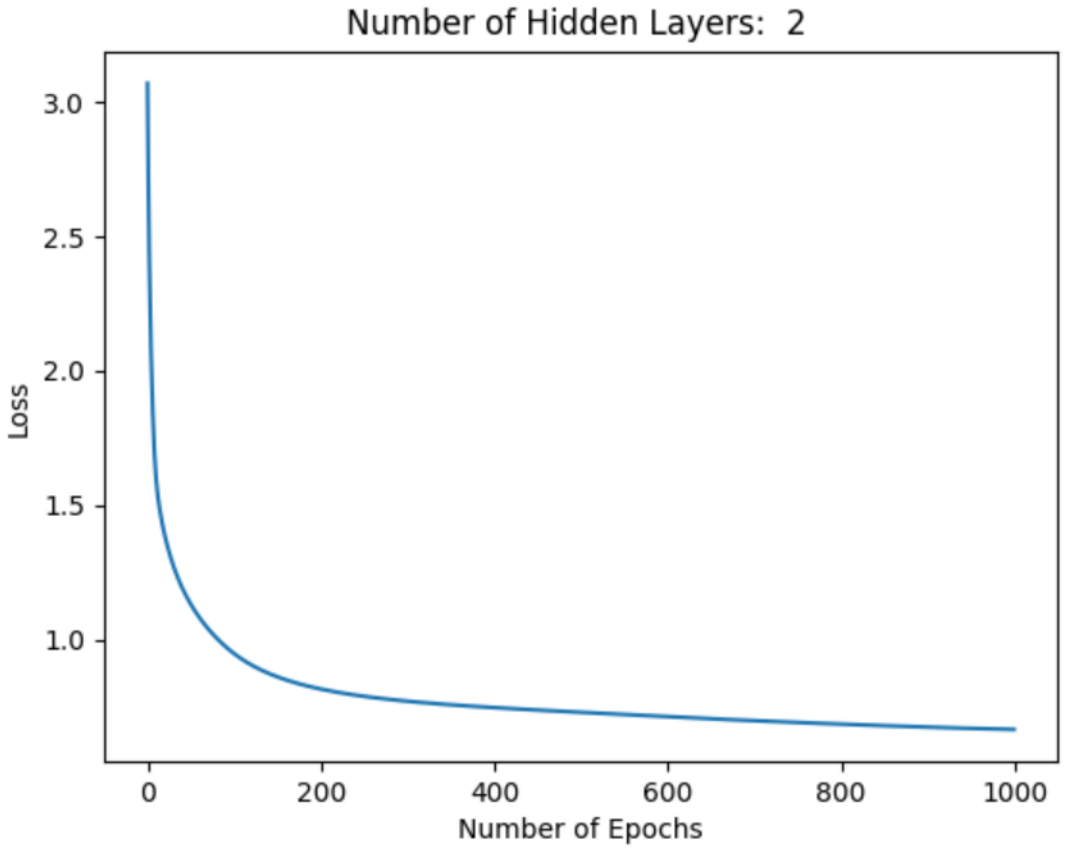
Number of Hidden Layers: 1
Training Accuracy 73.50%
Testing Accuracy 74.17%

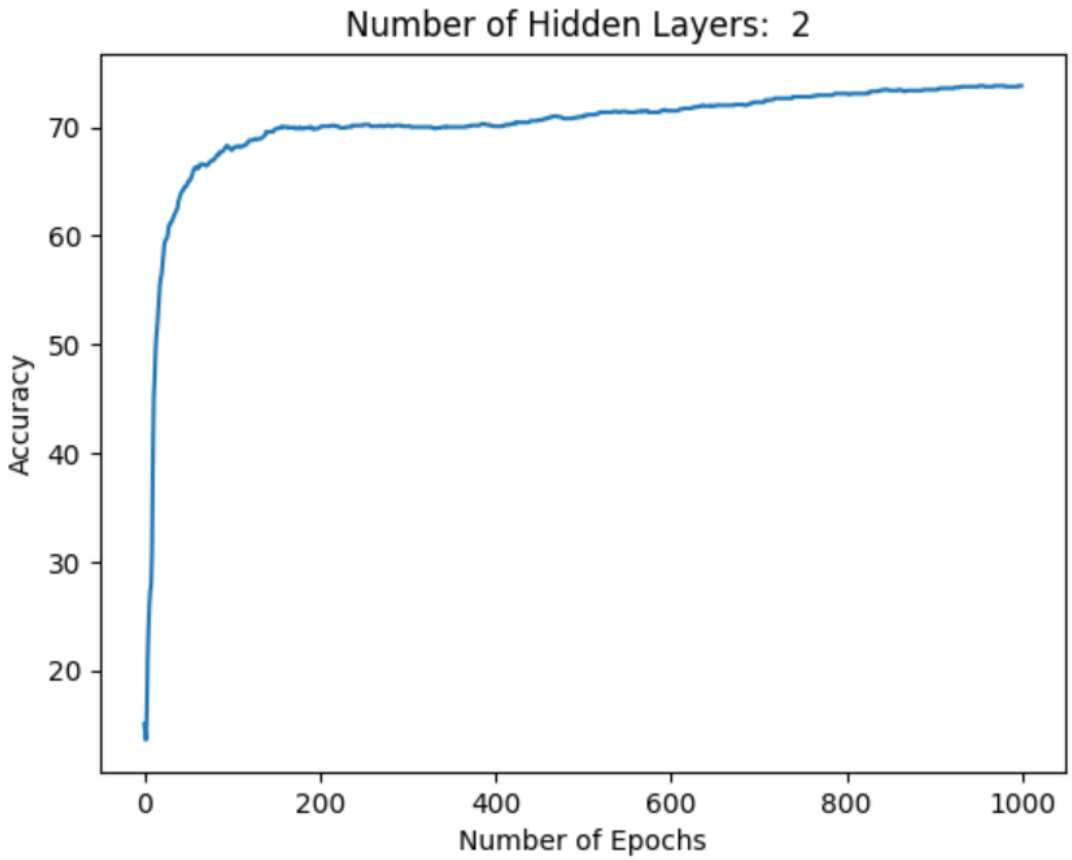
Number of Hidden Layers: 2
Training Accuracy 73.71%
Testing Accuracy 75.50%

4) **Plots of accuracy for three different learning rates for ANN with one hidden layer and two hidden layers.**


Number of Hidden Layers: 1
Loss
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0
200
400
600
800
1000
Number of Epochs


Number of Hidden Layers: 1
Accuracy
70
60
50
40
30
20
10
0
200
400
600
800
1000
Number of Epochs


Number of Hidden Layers: 2
Loss
3.0
2.5
2.0
1.5
1.0
0
200
400
600
800
1000
Number of Epochs


Number of Hidden Layers: 2
Accuracy
70
60
50
40
30
20
0
200
400
600
800
1000
Number of Epochs