DDA4210 Report

Zhizhen Chen 120090823

March 4, 2023

1 Main Procedure

The following are the main process that I used in this project:

1. Visualize the labels distribution and features distributions and perform PCA.
2. Split the dataset into training and validation set, and let the number of different labels equal in the training set.
3. Build up the bpnn using pytorch.
4. Tuning the model base on the accuracy of the validation set by choosing different hyperparameters.
5. Use the best model to predict the test data.

2 Model Construction

The feature analysis result shows that only feature 4 has the feature importance that is over 1, so weak classifier may cause inaccuracy. Due to the stronger generalization and error tolerance ability, I choose to build a two layers back-propagation neural network(BPNN) for better robustness.

In this model:

1. The ReLU function is applied in the activation layers
2. The Sigmoid function is applied in the output layers
3. Every layer is full-connected with linear transformation.
4. Choose BCE loss as the loss function, the formular is shown below

$$BCEloss = \frac{1}{N}\sum_{n=1}^{N} l_n$$

where $l_n = y_n log(x_n) + (1 - y_n) log(1 - x_n)$ is the loss of the nth sample.

In this algorithm, the normalized data will first be pass forward into the model for prediction, then do back propagation for optimization. Each time we use one batch of sample for training. The algorithm of BPNN training is shown below:

Algorithm 1 BPNN training

Input: trainloader, numepochs, model, lr
for epochs=1 to numepochs **do**
  for (input,label) in trainloader **do**
    1. Normalized the input using BatchNorm()
    2. Feed forward the normalized training example through the network and calculate the output of the network.
    3. Using the Binary Classification Error Loss to generate the error between the desired output and the actual output of the network.
    4. Backpropagate the error through the network and calculate the gradients of the weights and biases using the chain rule.
    5. Update the weights and biases by SGD optimzer.
  end for
  6. Calculate the accuracy of validation set until satisfy.
end for
Output: Trained Model

Here is the visualization of the BPNN:

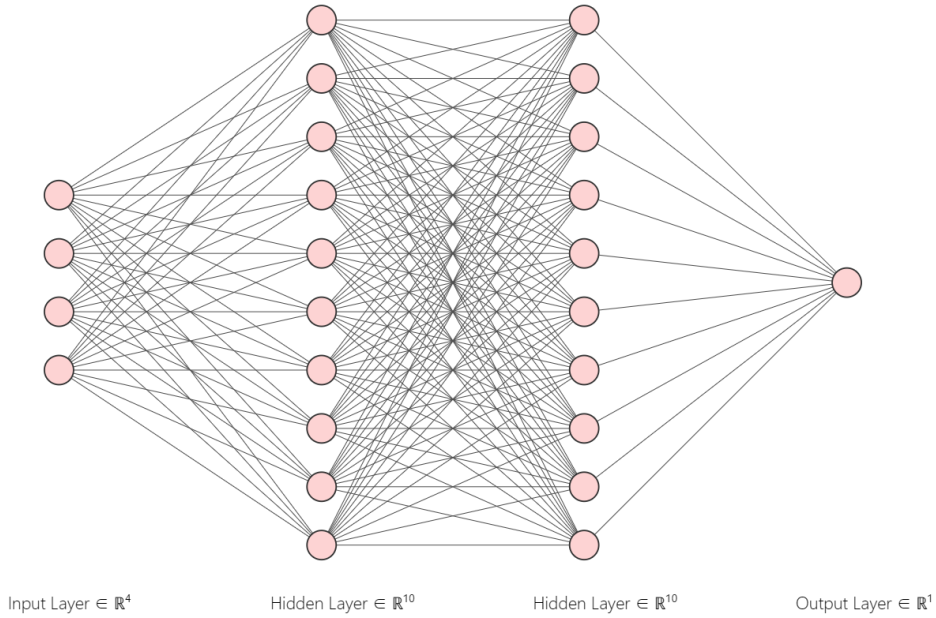

Figure 1: BPNN Visualization

3 Result

After tuning, the specific hyperparameters are chosen as follow to reach the highest accuracy:

Hyperparamter	Value
Hidden Layer 1	10
Hidden Layer 2	10
Epochs	6
Batch Size	128
Learning Rate	0.1

Table 1: Hyperparameters of BPNN

According to Kaggle result, this mdoel can reach to a 98.75% accuracy on the test data. The constructed model is attached in "120090823_model.pth ", the code is attached in "120090823_code.ipynb".