**Multilayer Perceptron**

A Multilayer perceptron is a feedforward artificial neural network consisting of at least three layers. Except the input node, each neuron consists of calculation of the following:

1. Matrix multiplication of weights and input data
2. Addition of biases
3. Application of a non-linear activation function

The assignment has used CIFAR-10 dataset which consists of 60000 images belonging to one of the ten classes.

**Preprocessing:**

Images are originally of 4D vector which consist of (samples, height, width, channels). They have been preprocessed to a 2D vector of (samples, pixels). The training and testing data consist of records each with 3072 pixels. Thus, the train data will be of shape (50000, 3072) and the test data will be of shape (10000, 3072). The data was further processed from uint8 to float so that we can divide the pixel values by 255 and get the pixel values in the range of 0 and 1.

**Model Definition:**

The model is defined in either 3 or 4 layers. An MLP with more than 4 layers just increased the number of parameters to be calculated and hence the time for execution resulting in almost the same accuracy. The model usually consisted of 1 input layer, 2 or 3 hidden layers and 1 output layer. The number of neurons for calculation varied from 2304 to 3072 neurons.

**Forward Propagation:**

The forward propagation consisted of defining weights and biases and also calculating the loss using categorical cross entropy loss function. Categorical Crossentropy function has been used because it is preferable with multiclass classification. Mean squared error and binary crossentropy can be used for performing linear regression and binary class classification. The activation functions varied from using tanh, relu, softmax and sigmoid. The idea of using softmax over sigmoid was figured out after noticing the difference in accuracy and loss. We reported softmax was beneficial for multiclass classification and sigmoid was better for binary class classification.

**Backward Propagation:**

This consist of defining the optimizer. The optimizer will specify the exact way in which the gradient of the loss will be used to update the parameters. Assignment included two types of optimizers - stochastic gradient descent and RMSprop. The learning rate usually varied between 0.001, 0.0001, 0.05 and 0.01. The learning rate defined the rate at which the model learned and it was found that 0.001 learning rate was the best one with RMSprop as the optimizer.

**Model Optimization:**

The model overfitted many times. Later it was found beneficial to try to overfit the data to get maximum training accuracy with trials and errors. Once the maximum training accuracy overfitted model is finalized, we can decide to choose over various optimization techniques to optimize our model so that we get maximum test accuracy as well. The optimization techniques varied from using L1 Regularization, L2 Regularization and Dropout techniques.

The following were the list of experiments performed:

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Models** | **Epochs** | **Batch Size** | **Number of layers** | **Number of neurons** | **Learning rate** | **Activation Functions** | **Dropout rates** | **Train Data** | **Test data** | **Train Accuracy** | **Test Accuracy** | **Result** |
| MLP\_1 | 20 | 128 | 3 | 3072 |  | relu,softmax | 0.2 | 10000 | 10000 | 0.098 | 0.1 |  |
| MLP\_2 | 20 | 512 | 3 | 3072 | 0.0001 | relu,softmax | 0.2 | 50000 | 10000 | 0.44 | 0.43 |  |
| MLP\_3 | 20 | 512 | 3 | 2304 | 0.001 | relu,softmax |  | 50000 | 10000 | 0.52 | 0.48 |  |
| MLP\_4 | 25 | 1024 | 3 | 2304 | 0.001 | relu,softmax |  | 50000 | 10000 | 0.46 | 0.42 |  |
| MLP\_5 | 30 | 512 | 3 | 2304 | 0.0001 | relu,softmax |  | 50000 | 10000 | 0.61 | 0.52 | Overfit |
| MLP\_6 | 30 | 1024 | 4 | 2688 | 0.001 | relu,softmax |  | 50000 | 10000 | 0.49 | 0.45 |  |
| MLP\_7 | 20 | 1024 | 4 | 2688 | 0.001 | relu,softmax | 0.2 | 50000 | 10000 | 0.41 | 0.43 | Overfit |
| MLP\_8 | 20 | 512 | 4 | 3456 | 0.001 | relu,softmax |  | 50000 | 10000 | 0.1 | 0.1 |  |
| MLP\_9 | 60 | 512 | 3 | 2304 | 0.0001 | relu,softmax |  | 50000 | 10000 | 0.73 | 0.53 |  |
| MLP\_10 | 100 | 512 | 3 | 2304 | 0.0001 | relu,softmax | L1-L2 | 50000 | 10000 | 0.89 | 0.76 |  |

**Result:**

MLP\_10 was found to be the best model with 0.76 accuracy.