Machine Learning Assignment 3

Chandragupta
M.Tech Computer Technology
2019EET2341

Abhishek Roy
M.Tech Computer Technology
2019EET2337

***Abstract*—This document has been submitted towards partial fulfillment of the course "Introduction to Machine Learning" taught by Dr Prathosh A.P during fall 2019 under the code ELL 784.**

I. MNIST FASHION DATASET

A. Dataset Description

Each image is 28 pixels in height and 28 pixels in width, for a total of 784 pixels in total. Each pixel has a single pixel-value associated with it, indicating the lightness or darkness of that pixel, with higher numbers meaning darker. This pixel-value is an integer between 0 and 255. The training and test data sets have 785 columns. The first column consists of the class labels, and represents the article of clothing. The rest of the columns contain the pixel-values of the associated image.

B. Artificial Neural Network

- Initialization of network - The network is initialized with a set number of inputs, hidden layers and outputs. Weights are randomly generated in the beginning and later updated during training.
- Over view - The model consists of feed forwarding the input to the output, then calculating the error between the expected value and the output of network, then back propagating the error till the input and updating the weights. Again repeating the same steps for the next sample in the training set.
- Feed-Forwarding
  - Activation - For each layer activate function takes all the outputs from the previous layer, multiplies with corresponding weights, sum it up to create the input for the neuron in the current layer.

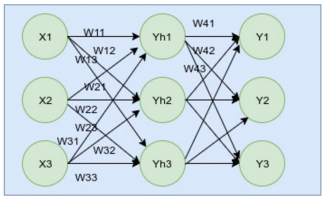


Fig. 1. Feed Forwarding

  - Activation Function - The Sigmoid function is used as activation function. For every neuron, activation is the input which is passed through the activation function to create the output.
- Back Propagation of error - For every neuron, a "delta" label is added in which the error is stored. The error is calculated using stochastic gradient method. The error stored under delta label are used during error calculation for next iteration.
- Updating the weights - This is online training. Hence the wights are updated for every sample of training data. Weights are updated for a given learning rate.
- Drop Outs - For each iteration drop out is used to randomly drop the activation of neurons in each layer. The activations are compared with a randomly shuffled binary array. The layer drops the neurons corresponding to all zeros.

C. Artificial Neural Network with Back-Propagation

We implement ANN with input layer which consist of 785 node, a hidden layer with variable node,and output with 10 node.we implemented back propagation from scratch.the lable of were integers from 1 to 10, we need to convert that ino one hot notation.For this we created a zero matrix with dimension Nx10 ,then the value of actual output is taken as index and narked as 1 in corresponding row of c mtrix.NN is dependent on the choice of no of nodes in hidden layer , initialization of weight matrix , learning rate and Epoch.we tried with different no of nodes, learning rate , epoch etc as shown in the table, with only difference in learning rate from 0.001 to 0.1 the accuracy drastically reduced to 9% from 74%.
Loss function:

$$E(W) = \Sigma\Sigma t_{nk} * log(y_{nk})$$

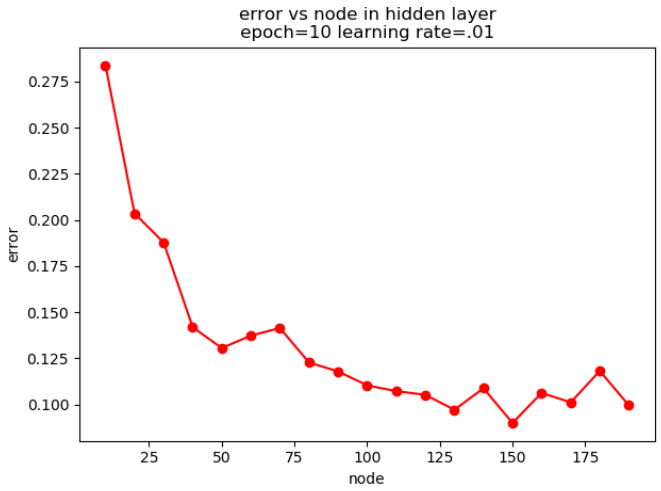
Stochastic Gradient decent

$$W^(\tau + 1) = W^{\tau} - \eta \nabla_w E$$

Nodes	iteration	L$_r ate$	Accuracy
20	10	.01	83
100	10	.01	88
50	10	.1	9
50	10	.001	74

D. Artificial Neural Network : Back propagation with drop out

The basic idea of Drop-Out is to reducing over fitting in neural networks by preventing complex co-adaptations on



The above figure has been plotted to show the variation of error with respect to the number of nodes in the hidden layer.As we can see, the error is decreases, on an average, as the number of neurons in the hidden layer increases. As the number of neurons increases, there are also chances of over fitting. To avoid that, we have used cross validation to split the data instead of simple train-test split.

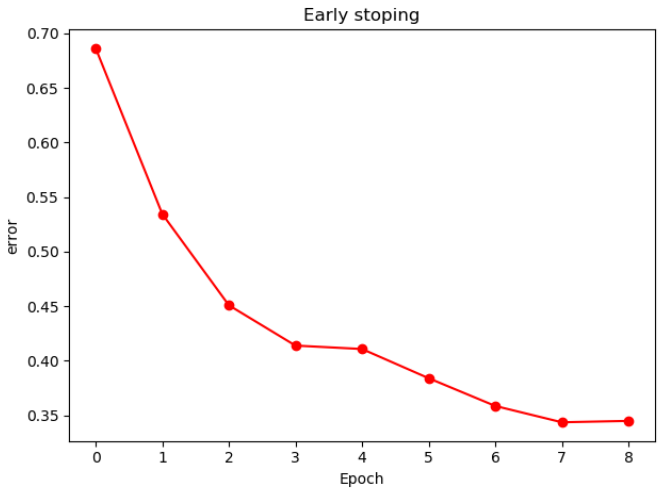
training data.to implement dropout in neural network we initializes array which consist of 1 and 0 according to drop percentages.then we set a_ij to 0 by index of created array.

Drop%out	Nodes	Accuracy	Epochs
20	100	81	10
50	100	63	10
75	100	57	10
75	100	70	100

As we can see with increase in drop percentage increase error , this is due to constant Epochs, when we implement dropout we have to increase the no of epochs.As shown in the last column of table, increasing the no epochs 10 to 100 our accuracy increase from 57 percent to 70 percent.So we conclude that larger the dropout the less accuracy and to compensate it we need to increase no of epochs.The main benifit of using drop out is that it may perform worst on train data but give good result on test data

E. Artificial Neural Network with Back-Propagation and early stopping

The figure 2 shows the condition of early stopping. With every epoch, the error decreases. After the 7th epoch, the error starts to increase and hence the training is stopped.At the same time we maintain a copy of previous weigth matrix, when we found increase in error we terminate the loop and update the weight matrix to previous copy.



Red - Validation Error. Blue - Test Error

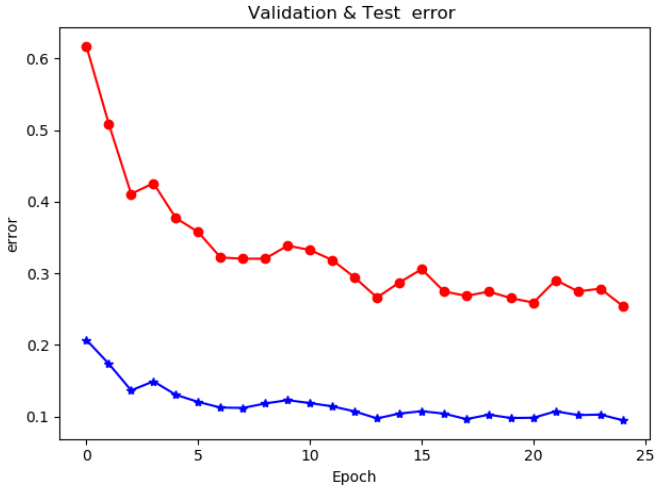


Fig. 4. Validation and test error vs Epoch.

The above picture captures the Validation and test error after every epoch.

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1

Fig. 5. First Five Data

II. Heart Disease Dataset

A. Dataset Description

B. Method used : Artificial Neural Network

- Splitting of data - Data is split into 5 subsets using cross-validation. This reduces the chances of over fitting during training.
- Epoch and learning rate - The model is trained with epoch 300 and learning rate of 0.09 is used for stochastic gradient method.
- Back propagation - Errors calculated at each iteration are propagated back wards and stored in the neurons under the ”delta” label which are later used for updating the weights.

Accuracy - THe mean accuracy achieved with this dataset using ANN is 81.33 percent.

learning rate	accuracy
0.3	83.33
0.9	80.00
0.09	81.00

Number of Epochs	Accuracy
500	81
20	77.33
200	82.67

III. Convolutional Neural Network

- Data-set - We modeled CNN on MNIST Image Dataset and used keras library to build the model.
- Architecture - It consits of following layers in the order:
  - 2 Convolutional Network layers - Filters which goes through width and length of Input matrix.
  - Pooling layer - For reduction of spatial size using dimensionality reduction. Here we use MAX pooling.
  - Drop out filter
  - Flatten
  - Dense

```
Epoch 1/3
60000/60000 [==============================] - 140s 2ms/step - loss: 2.3331 - acc: 0.1121
Epoch 2/3
60000/60000 [==============================] - 138s 2ms/step - loss: 1.1048 - acc: 0.6616
Epoch 3/3
60000/60000 [==============================] - 139s 2ms/step - loss: 0.4974 - acc: 0.8480
```

Fig. 6. Accuracy after each epoch

The above picture captures the variation of accuracy with epoch for sigmoidal activation function in the first two convolving layers.

```
Epoch 1/3
60000/60000 [==============================] - 138s 2ms/step - loss: 0.7167 - acc: 0.7794
Epoch 2/3
60000/60000 [==============================] - 137s 2ms/step - loss: 0.2254 - acc: 0.9350
Epoch 3/3
60000/60000 [==============================] - 137s 2ms/step - loss: 0.1360 - acc: 0.9612
```

Fig. 7. Accuracy after each epoch

The above picture captures the variation of accuracy with epoch for ReLU activation function in the first two layers.

```
Epoch 1/3
60000/60000 [==============================] - 141s 2ms/step - loss: 0.6099 - acc: 0.8198
Epoch 2/3
60000/60000 [==============================] - 140s 2ms/step - loss: 0.2174 - acc: 0.9376
Epoch 3/3
60000/60000 [==============================] - 140s 2ms/step - loss: 0.1440 - acc: 0.9586
```

Fig. 8. Accuracy after each epoch

The above picture captures the variation of accuracy with epoch for tanh activation function in the first two layers.

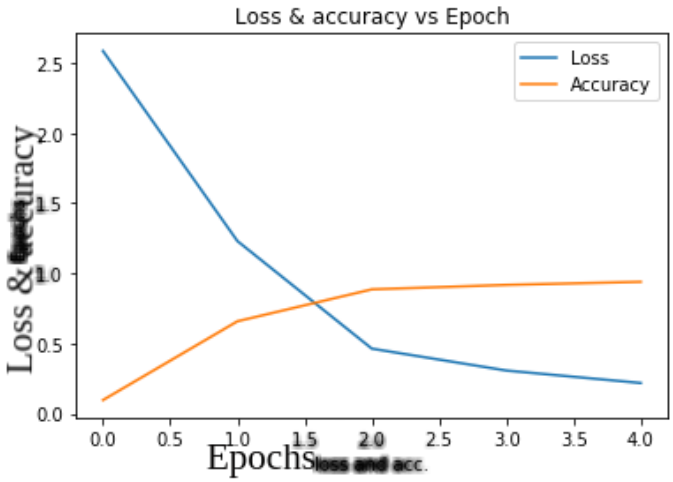


Fig. 9. Accuracy after each epoch

the above graph shows how loss and accuracy changes with choice of epoch(after 3 epochs,there is no much benefit, epoch greater than 4 will be waste of calculation)

Acknowledgment

We would like to thank Dr Prathosh A.P for his continuous guidance and motivation to keep learning even after the failed attempts. We have used kaggle for data sets.

.