Machine Learning Assignment 4

Sayan Sinha
16CS10048

INTRODUCTION

Artificial Neural Networks are the computational models inspired by the human brain. Many of the recent advancements have been made in the field of Artificial Intelligence, including Voice Recognition, Image Recognition, Robotics using Artificial Neural Networks. The term 'Neural' is derived from the human (animal) nervous system's basic functional unit 'neuron' or nerve cells which are present in the brain and other parts of the human (animal) body. A neural network is a group of algorithms that certify the underlying relationship in a set of data similar to the human brain. Mathematically speaking, the neural network is a function approximator. More the number of layers, the higher dimensional curve you get. More the number of neurons, the more intersecting curves of each dimension is made available to represent the given function.

MODEL

Part 1

The model features a neural network with 500 input neurons which represent 500 most frequently used words in the corpus. The hidden layer has 100 neurons and is activated by ReLU. The output has a single node and is activated by sigmoid. ReLU is defined as:

$$f(x) = x^+ = \max(0, x)$$

whereas Sigmoid is given as:

$$S(x) = \frac{1}{1 + e^{-x}} = \frac{e^x}{e^x + 1}.$$

The derivative of ReLU is as simple as it being 0 if the input was less than 0, and 1 otherwise. The derivative of sigmoid is given as:

$$f'(x) = \frac{\sigma(x) - 1}{\sigma(x)}$$

Part 2

The second part of the assignment had two hidden layers whose values were to be given as user inputs. In all experimentations, however, both were assumed to possess 100 neurons each. The hidden layers were activated by sigmoid whereas the output, which this time had two nodes (multi-class classification model) was activated by softmax in order to convert them to probabilistic values. Softmax is defined as:

$$S_j = \frac{e^{a_j}}{\sum_{k=1}^{N} e^{a_k}} \qquad \forall j \in 1..N$$

The derivative of Softmax is given by:

$$D_j S_i = \begin{cases} S_i(1 - S_j) & i = j \\ -S_j S_i & i \neq j \end{cases}$$

Cross-entropy loss was used to measure the cost in both the experiments. Both were optimised using Stochastic Gradient Descent, i.e. the mini-batch size of the algorithm was set to unity.

EXPERIMENTATION

Part 1

Table 1

Iters	Train error	Test error
500	0.1406006147	0.08194866196
1000	0.04740744121	0.08094866195
1500	0.03051498202	0.07994866194
2000	0.02055279631	0.07894866153
2500	0.01639457649	0.07794866236
3000	0.01473185189	0.07694866196
3500	0.01199972726	0.07594866195
4000	0.009366965462	0.07494866193
4500	0.007953811747	0.07394742493
5000	0.00719751382	0.07294864539

The table has been plotted in figure 1 for proper visualisation. The blue line represents training error, the red line shows the test error. The y-axis shows error and x-axis gives the iteration when increased 600 times.

The final train accuracy was 99.798% and test accuracy was 98.296%.

An interesting observation was noticed in this experiment. There is a huge data imbalance in the given dataset (approximately 43:7). So, when an attempt was made to reduce the "ham" classified emails by 80%, there was a drop in accuracy. But, in the case when the "bias" of a neural network was taken out, there were drastic improvements. For instance, the test accuracy was only 47.14% with 50% reduction which increased to 69.23% with 80% reduction.

Part 2

Table 2

Iters	Train error	Test error
500	1.464558537	0.3964909603
1000	0.9492836087	0.3964389589
1500	0.4028325523	0.3960384389
2000	0.2652494697	0.3959437283
2500	0.1993061368	0.3957348329
3000	0.158942243	0.3956439284
3500	0.1304699749	0.3953853298
4000	0.1089801127	0.3951832849
4500	0.09216003919	0.3948324923
5000	0.07882565188	0.3964743284

A similar figure has been shown in figure 2 representing the above table. The final train accuracy was 98.941% and test accuracy was 95.211%.

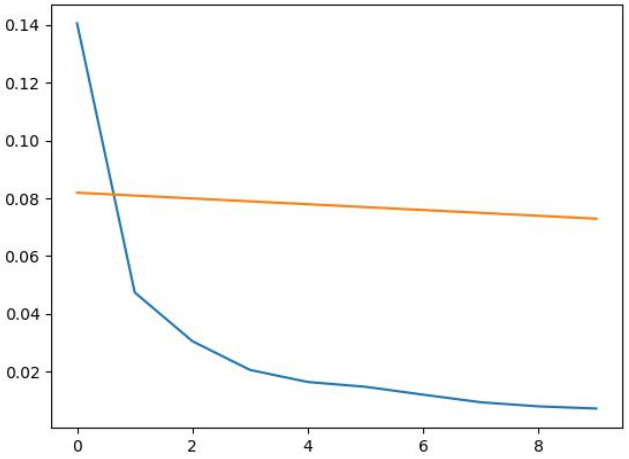

Fig 1

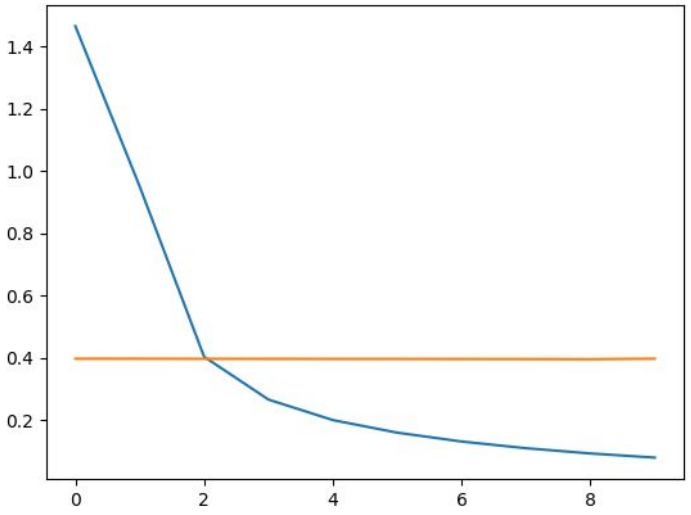

Fig 2

CONCLUSION

The accuracy achieved was satisfactory. Hence this experiment is used to show that neural networks are a powerful tool in machine learning. Data imbalance, however, is a negative aspect, as this was a supervised learning problem. Proper statistical data imbalance removal methods should be followed in case data imbalance removal is attempted.

REFERENCES

1. LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. "Deep learning." nature 521.7553 (2015): 436.
2. Le, Quoc V., Navdeep Jaitly, and Geoffrey E. Hinton. "A simple way to initialize recurrent networks of rectified linear units." arXiv preprint arXiv:1504.00941 (2015).