NLP - Mini 2

Sentiment Analysis

1 Introduction

The project involved implementation of neural networks for the task of Sentiment Analysis on the Rotten Tomatoes data set which consists of sentences classified as either positive or negative sentiment. A simple feed forward neural network using deep averaging and a recurrent neural network was implemented on PyTorch. For the task of supervised learning, deep learning eliminates the need to do feature engineering by translating the data into compact intermediate representations where each layer in the neural network picks out the features that improve performance. More layers in the deep model help in learning the features effectively. For a feed forward neural network, comprises of many hidden layers and one output layer. For recurrent neural networks, in which a signal may propagate through a layer more than once, the depth is potentially unlimited.

2 Model

All the input sentences are padded with zeroes to make them of equal length and for each input sentence, all the words are represented as 300 dimensional GloVe vectors which is fed to the input layer of the network. A batch size of 5 was used, with learning rate of 0.01 and the models were trained for 5 epochs each with the final output layer being a sigmoid function since this is a binary classification problem. For the weight update, Adam optimizer and binary cross entropy loss function was used. The initial weights for the model are assigned using Xavier initialization with uniform distribution.

2.1 Feed Forward Neural Network

For each sentence, the average of all the word embeddings is fed as input to the first layer in the network. The output of the input layer goes through two hidden layers with 80 neurons followed by a dropout layer with dropout probability of 0.3. The final layer of the feed forward neural network is a sigmoid layer and the last column is considered to be the predicted output labels.

2.2 Recurrent Neural Network

The model comprises of a single layered, unidirectional LSTM. The output of the LSTM layer is fed to a hidden layer of size 80 with dropout of 0.5. Rather than taking the output of only the last time step in the final sigmoid layer, the average across all the time steps seemed to give better accuracy.

3 Challenges

In deep neural networks, hyper parameter tuning is a big challenge. Reducing the batch size helped the model to learn better and somehow bidirectional LSTM with more layers didn't seem to perform as good as a single layered uni directional LSTM. Training for more epochs resulted in over-fitting and reduced accuracy on the validation set. Increases the neurons in the hidden layers also seemed to increase efficiency as the number of features learnt increases proportionally.

Table 1: Epochs vs Accuracy

Dimensions	Epochs	Accuracy
50	1	68.0
	2	67.7
	3	69.5
	4	69.6
	5	69.7
300	1	74.9
	2	72.9
	3	75.4
	4	73.5
	5	75.5

4 Conclusion

The accuracy of the feed forward neural network for different epochs and by using the 50-dim and 300-dim was analysed and the results have been tabulated in Table 1. As the weight updates are done per batch, we get a decent accuracy even in one epoch. A maximum accuracy of 75.5% and 76.9% was achieved on the development set by training the model using feed forward and recurrent neural network respectively. Training for more than 5 epochs resulted in over-fitting and there was a sudden drop in the validation loss.

Collaborator - Madhumitha Sakthi