Sarcasm Detection in News Headlines

BITS Pilani Hyderabad Campus CS F429 Natural Language Processing Project

By

Akshat Jain

Arjun Prasad

Aryan Saraf

Pranav Sistla

f20190117@hyderabad.bits-pilani.ac.in

f20190183@hyderabad.bits-pilani.ac.in

f20190138@hyderabad.bits-pilani.ac.in

f20191220@hyderabad.bits-pilani.ac.in

Abstract

NLP frequently uses sentiment analysis to better comprehend people's subjective viewpoints. However, if people use sarcasm in their statements, the results of the analysis may be skewed. It is vital to be able to identify sarcasm in order to correctly grasp people's genuine intentions. Many prior models for detecting sarcasm were built solely on the basis of isolated utterances, i.e. just the reply text. With special emphasis, LSTMs were used as the major models. We hope to improve the accuracy of sarcasm identification in this study by looking into the importance of contextual information in identifying sarcasm. We ran multiple LSTM models including some with a CNN layer and a BERT model. We observed that BERT gave us better accuracy as compared to LSTM models.

Introduction

Sarcasm is the use of language that normally signifies the opposite in order to mock or convey contempt. Sarcasm detection is a very narrow research field in NLP, a specific case of sentiment analysis where instead of detecting a sentiment in the whole spectrum, the focus is on sarcasm. Therefore, the task of this field is to *detect if a given text is sarcastic or not.* The first problem we come across is that, unlike in sentiment analysis where the sentiment categories are very clearly defined (love objectively has a positive sentiment, hate a negative sentiment no matter who you ask or what language you speak), the borders of sarcasm aren't that well defined. People of different genders, castes and racial backgrounds perceive sarcasm very differently. Also, sarcasm is highly subjective, and depends a lot on the person perceiving it.

In this project, we considered three NLP models. The first one is a vanilla Bidirectional LSTM model. The second one is a CNN+LSTM model and the third one is a BERT based classifier. We tried out best to avoid overfitting and tuned hyperparameters to the best extent possible. We also tried varying the number of epochs, adding regularization, number of hidden units to determine the impact on sarcasm detection performance.

Long Short-Term Memory networks (LSTMs) are a special kind of RNN, capable of learning long-term dependencies. LSTMs have been used for performing sentiment analysis for a long time and have performed exceedingly well.

The BERT model took a long time to train but performed really well and gave the best accuracy. With very little hyper-parameters tuning, BERT clearly exhibits the benefits of utilizing pretraining models to obtain state-of-the-art results.

BERT (Bidirectional Encoder Representations from Transformers) is a recent paper published by researchers at Google AI Language. BERT makes use of Transformer, an attention mechanism that learns contextual relations between words (or sub-words) in a text. In its vanilla form, Transformer includes two separate mechanisms — an encoder that reads the text input and a decoder that produces a prediction for the task. Since BERT's goal is to generate a language model, only the encoder mechanism is necessary. As opposed to directional models, which read the text input sequentially (left-to-right or right-to-left), the Transformer encoder reads the entire sequence of words at once. Therefore, it is considered bidirectional, though it would be more accurate to say that it's non-directional. This characteristic allows the model to learn the context of a word based on all of its surroundings (left and right of the word).

Related Work/Citations

1) Sarcasm Detection: Lydia Xu, Vera Xu - 15791781.pdf (stanford.edu)

This research paper applies mainly two models LSTM and BERT. The LSTM implemented include vanilla LSTM, Conditional LSTM and LSTM with sentence-level attention. The models were applied on two different datasets namely discussion forum data and reddit data. The results of this paper show that BERT outperforms all the LSTM the reason of which could be informal language used in reddit data.

2) Deep CNN-LSTM with word embeddings for News headline Sarcasm Detection: Paul Mandal, Rakeshkumar V Mahto

This paper works on a news headlines dataset which is more clean data as there will be less use of informal language compared to reddit data. The architecture that proposed in this paper consists of an embedding layer, a CNN, and a bidirectional LSTM on word-level vector encodings of news headlines. This network architecture leverages advantages of the LSTM and CNN. This deep CNN-LSTM with word embedding was able to achieve an accuracy of 86.16%.

3) Sarcasm Detection on Twitter: A Behavioural Modelling Approach Ashwin Rajadesingan, Reza Zafarani, and Huan Liu

This paper works on a dataset consisting of tweets from twitter. In this paper, the author's introduce SCUBA, a behavioural modelling framework for sarcasm detection. They discuss different forms that sarcasm can take, namely: (1) as a contrast of sentiments, (2) as a complex form of expression, (3) as a means of conveying emotion, (4) as a possible function of familiarity and (5) as a form of written expression. This paper got an accuracy of 79.38%.

4) Context-Aware Sarcasm Detection Using BERT: Arup Baruah , Kaushik Amar Das , Ferdous Ahmed Barbhuiya, and Kuntal Dey

This paper works on two different datasets namely twitter dataset and reddit dataset. This paper uses 3 models including BiLSTM, BERT and SVM. The authors found that including context in the form of the last utterance in a dialogue chain slightly improved the performance of the BERT classifier for the Twitter data set compared to just using the response alone. For the Reddit data set, including the context did not improve the performance.

5) Sarcasm Detection using Hybrid Neural Network: Rishabh Misra, Prahal Arora

To overcome the limitations related to noise in Twitter datasets, the authors collected a new Headlines dataset from two news website one of which is TheOnion which aims at producing sarcastic versions of current events. This new dataset has following advantages over the existing Twitter datasets:

• Since news headlines are written by professionals in a formal manner, there are no spelling mistakes and informal usage. This reduces the sparsity and also increases the chance of finding pre-trained embeddings.

• Furthermore, since the sole purpose of TheOnion is to publish sarcastic news, we get high quality labels with much less noise as compared to twitter datasets.

• Unlike tweets which are replies to other tweets, the news headlines are self-contained. This would help in teasing apart the real sarcastic elements.

6) Tweet Sarcasm Detection Using Deep Neural Network: Meishan Zhang, Yue Zhang and Guohong Fu

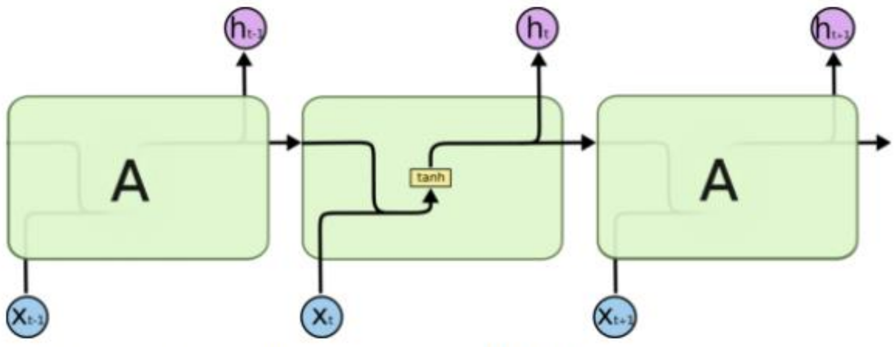
This paper uses the neural model with only local features to evaluate the effect of different word embedding initialization methods. A better accuracy is obtained by using GloVe embeddings for initialization compared with random initialization.

Approach/Methodology

LSTMs are a modified form of RNNs which do not suffer from the vanishing gradient/exploding gradient problem enabling us to remember past information for a long time.

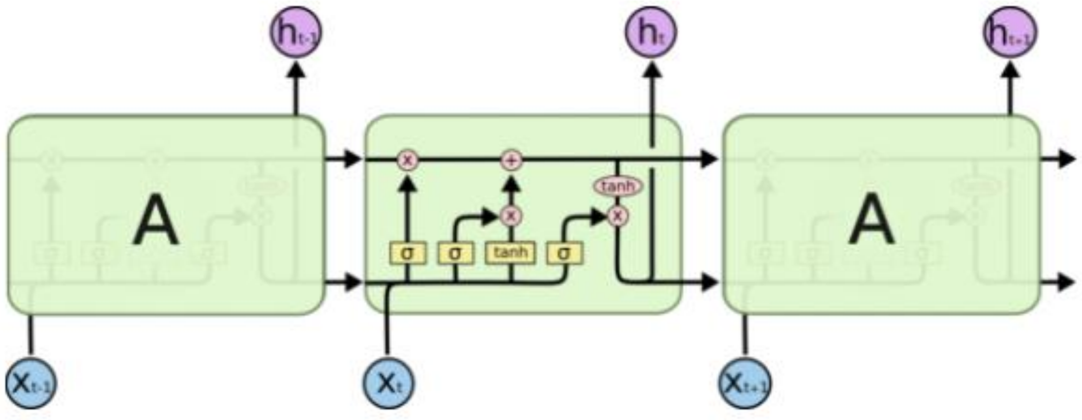
Sometimes, we only need to look at recent information to perform the present task and, in these cases, RNNs are extremely useful but there are also cases when more context is required. Unfortunately, as that gap grows, RNNs become unable to learn to connect the information.

All recurrent neural networks have the form of a chain of repeating modules of neural network. In standard RNNs, this repeating module will have a very simple structure, such as a single tanh layer.



The repeating module in a standard RNN contains a single layer.

LSTMs also have this chain like structure, but the repeating module has a different structure. Instead of having a single neural network layer, there are four, which interact enabling us to remember data for a longer time.



We implemented a vanilla LSTM, CNN+LSTM and a BERT network.

Vanilla LSTM

This is a simple bidirectional LSTM model. There is a bidirectional LSTM layer containing 64 nodes. Global Max Pool layer follows the LSTM layer. A Dense neural network was implemented sequentially consisting of 2 hidden layers (activation function is 'ReLU') having 64 and 32 nodes respectively. A dropout layer of 0.4 was also implemented after both the hidden layers. The final output layer has Sigmoid as the activation function as we have to do Binary Classification.

CNN+LSTM

This model comprises of a CNN layer followed by a bidirectional LSTM layer. The CNN layer detects 128 properties and has ReLU as the activation function. The LSTM layer has 64 nodes followed by a dense neural net containing 1 hidden layer. The hidden layer has 100 nodes and contains L1 regularization. Dropout layer of 0.2 has been used. The final output layer has a single node and activation function Sigmoid has been used as we have to do binary classification.

BERT

This model uses a BERT pretrained encoder called ‘bert-case-uncased’. A neural network consisting of 1 hidden layer has been implemented on top of the BERT encoder. The hidden layer consists of 128 nodes (activation function ‘ReLU). A dropout layer of 0.2 has also been implemented. The output layer has a single node and activation function of sigmoid has been used as we have to do binary classification.

Experiments

1) Dataset

The data is collected from two news websites. *TheOnion* aims at producing sarcastic versions of current events and we collected all the headlines from News in Brief and News in Photos categories (which are sarcastic). We collect real (and non-sarcastic) news headlines from *HuffPost*.
This new dataset has following advantages over the existing Twitter datasets:

- Since news headlines are written by professionals in a formal manner, there are no spelling mistakes and informal usage. This reduces the sparsity and also increases the chance of finding pre-trained embeddings.
- Furthermore, since the sole purpose of *TheOnion* is to publish sarcastic news, we get high-quality labels with much less noise as compared to Twitter datasets.
- Unlike tweets which are replies to other tweets, the news headlines we obtained are self-contained. This would help us in teasing apart the real sarcastic elements.

2) Evaluation method/Metrics

We evaluate all our models on the basis of the following parameters obtained during the test phase.

Accuracy

Accuracy is the ratio of current prediction to the total number of input samples.

$$Accuracy = \frac{Number\ of\ Correct\ predictions}{Total\ number\ of\ predictions\ made}$$

Precision

Precision is the number of correct positive results divided by the number of positive results predicted by the classifier.

$$\text{Precision} = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

Recall

Recall is the number of correct positive results divided by the number of ***all*** relevant samples (all samples that should have been identified as positive).

$$\text{Recall} = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

F1 Score

F1 Score is the Harmonic Mean between precision and recall. The range for F1 Score is [0, 1]. It tells you how precise your classifier is as well as how robust it is.

$$\text{F1} = 2 \times \frac{Precision * Recall}{Precision + Recall}$$

3) Experimental Setup

To increase size of dataset, we append the datasets Sarcasm Detection and Saracasm_detection_v2. This gives a corpus of 55,528 news headlines. To evaluate model performance, we first create a test set, that we keep alien from the training set. We then evaluate the model on the test set.

Size of test set = 11,066 (20% of 55,328)

Size of training set = 44,262 (80% of 55,328)

We then train the model on the training set with a cross validation split of 0.2

LSTM model:

Apart from running simple LSTM model, we also check performances with addition of dropout regularization

Experiment conditions:

Number of epochs = 30

Batch size = 256

Optimizer = Adam

Learning rate = 0.01

Loss function = binary cross entropy

We note that dropout regularization, reduces the problem of overfitting, and provides better performance on both the validation and test data.

CNN + BiLSTM model:

We assume a length of 32 words for each sentence. Reason for this assumption:

During EDA (exploratory data analysis), we found that 99.99% of the sentences are <= 25 words in length. Hence assuming that the max length for a sentence could be ~ 3 words before or after the likely sentence length, we can assume that the length of a sentence could be 32 words.

Embeddings used:

Simple embeddings layer, Glove 100d embeddings vectors

Regularization techniques used:

Dropout regularization, L2 regularization

Experiment conditions:

Number of epochs = 10

Batch size = 32

Optimizer = Adam

Learning rate = 0.01

Loss function = binary cross entropy

BERT model:

Through trial and error, we notice that the performance of the BERT model is best assuming a sentence length of 20 words.

We noted that the performance of the model shot up (test accuracy: 92.34% inc. to 97.54%) on increasing data (data was increased by appending Sarcasm_detection_v2 dataset to the original Sarcasm_Detection.json dataset)

Experiment conditions:

Number of epochs = 3

Batch size = 32

Optimizer = Adam

Learning rate = 1e-5

Loss function = binary cross entropy

Results and Discussion

Vanilla LSTM

We implemented 2 versions of Vanilla LSTM Models. One with dropout and one without dropout.

Without dropout:

- Train Accuracy = 99.85%
- Validation accuracy = 92.251%
- Test accuracy = 94.26%

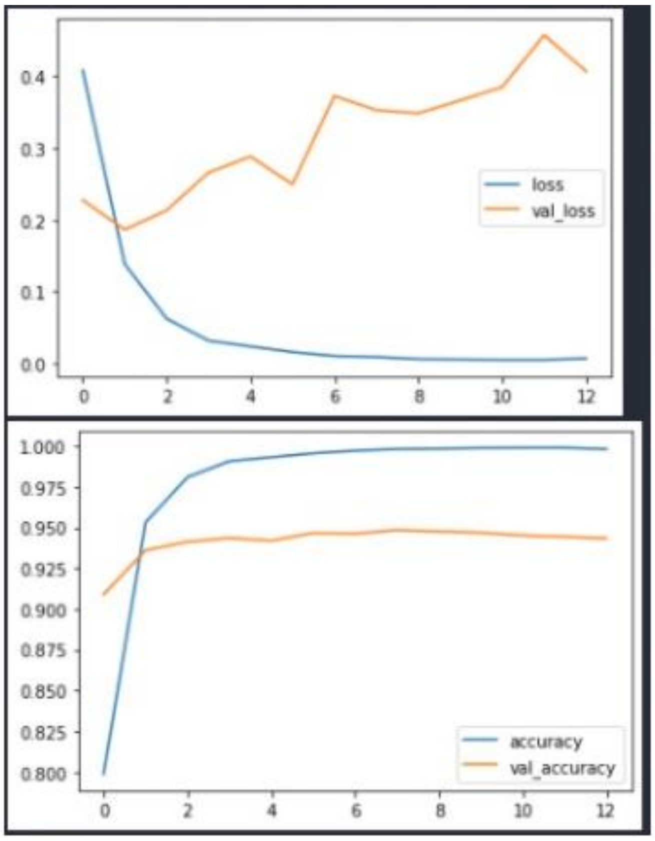
With dropout:

- Train accuracy = 99.81%
- Validation accuracy = 94.33%
- Test accuracy = 95.12%

Thus, dropout reduces the problem of overfitting and gives higher validation and test accuracies.

Final Results (with Dropout):

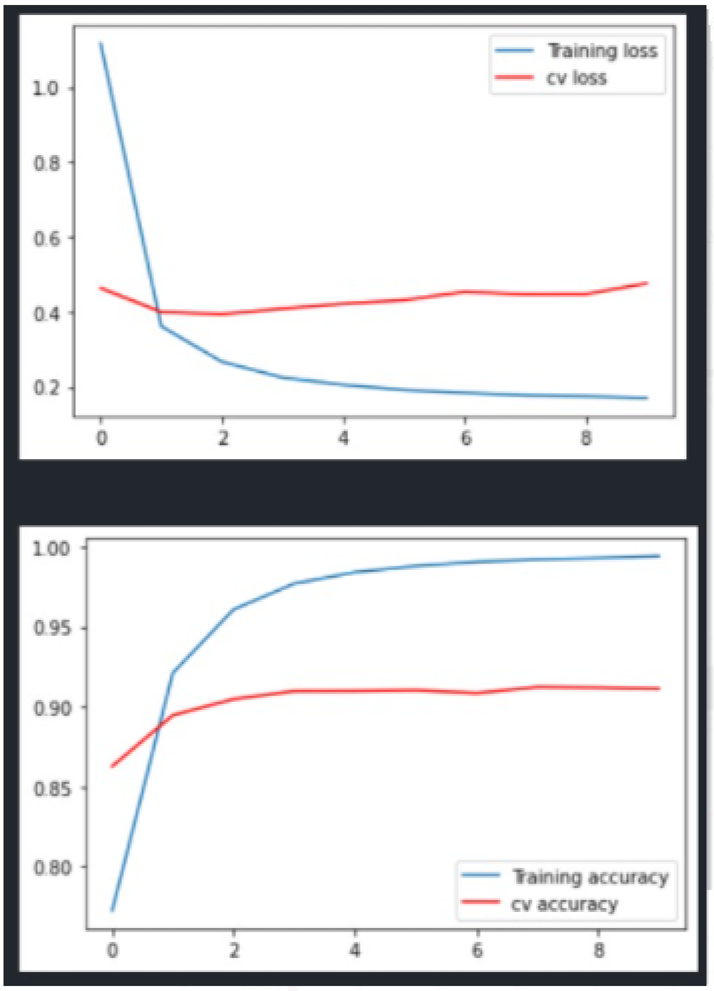
- Train accuracy = 99.81%
- Validation accuracy = 94.33%
- Test accuracy = 95.12%
- Test recall = 94.87%
- Test precision = 94.72%
- Test f1 score = 94.79%



CNN+BiLSTM

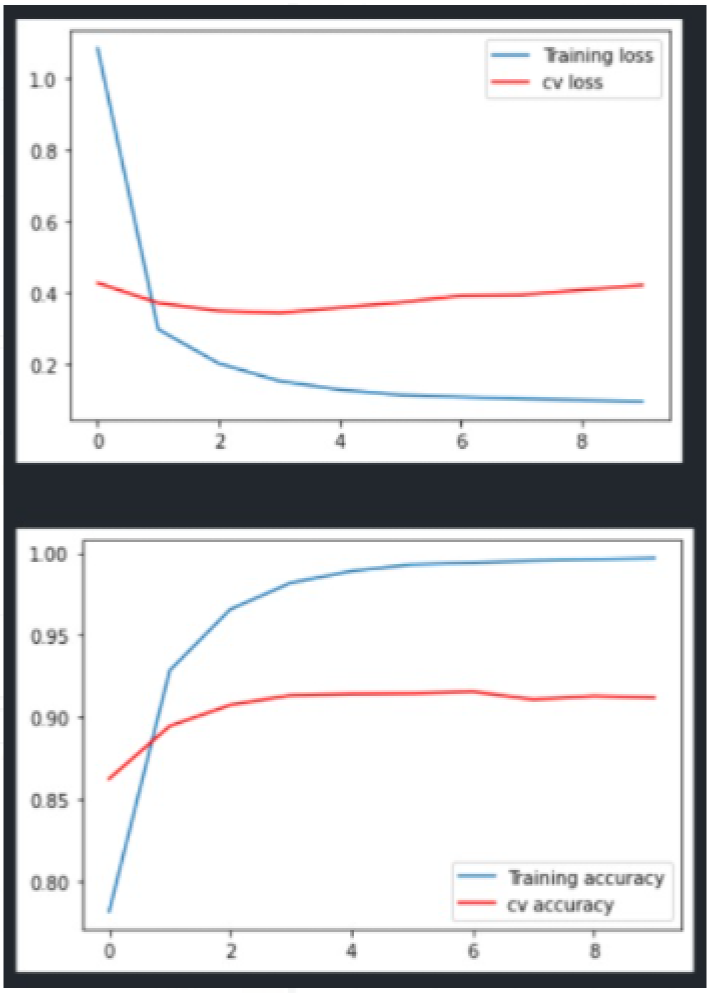
We implemented 4 variations whose results are documented as follows.

- Using dropout regularization and simple embeddings layer:
- Train accuracy = 99.40%
- Cross validation accuracy = 91.12%
- Test accuracy = 90.76%
- Test recall = 89.94%
- Test precision = 89.87%
- Test f1 score = 89.91%



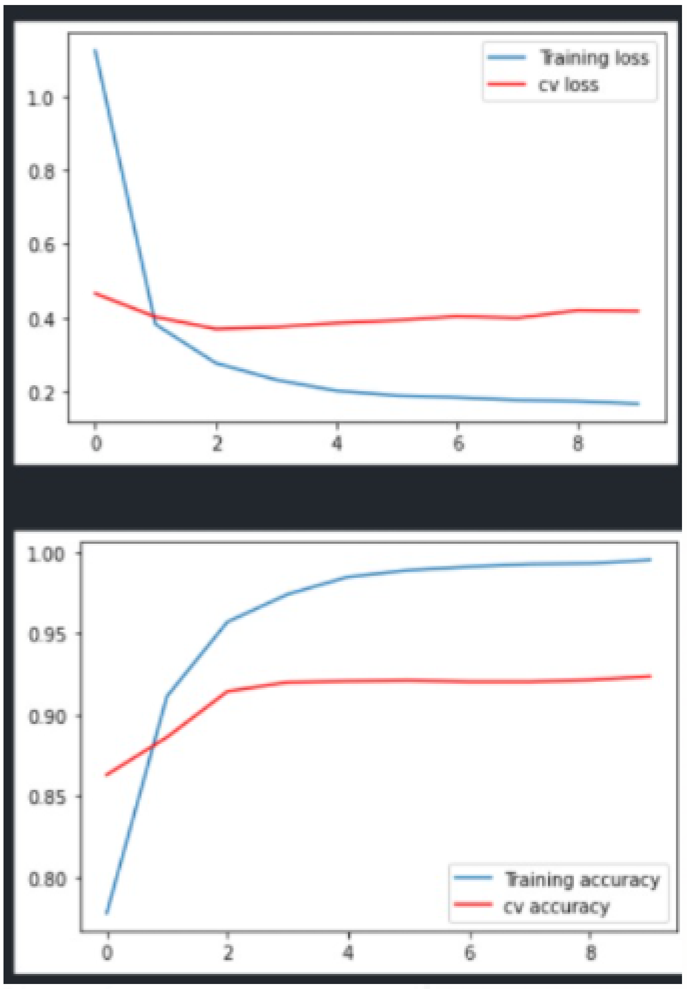
dropout + normal embedding

- Using no dropout regularization and simple embeddings layer:
- Train accuracy = 99.71%
- Cross validation accuracy = 91.20%
- Test accuracy = 90.81%
- Test recall = 90.86%
- Test precision = 89.25%
- Test f1 score = 90.04%



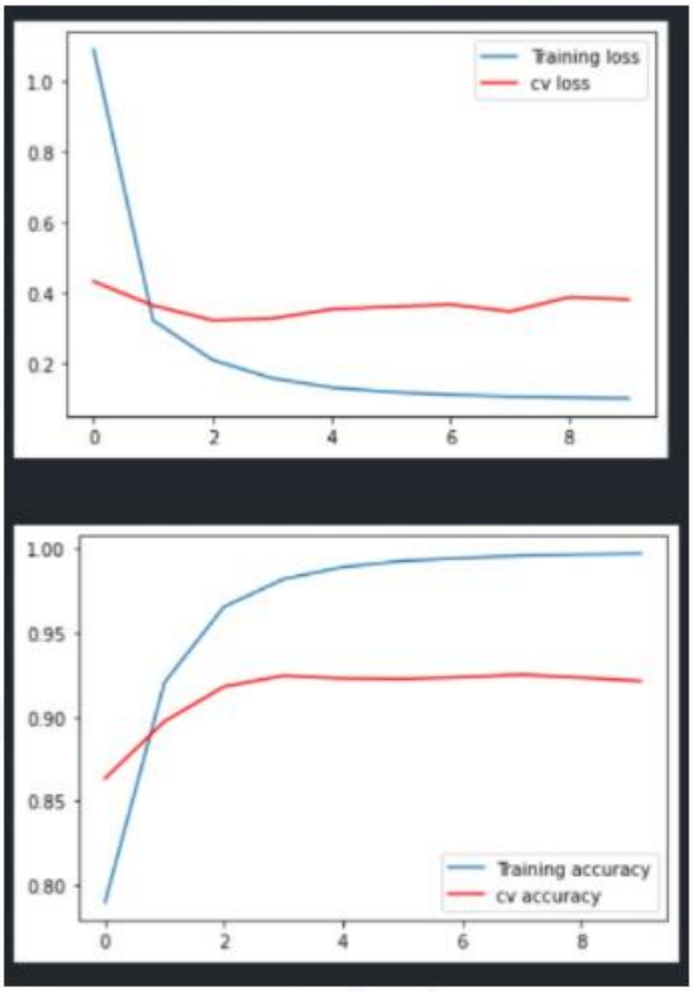
no dropout + normal embedding

- Using dropout regularization and GloVe 100d embeddings layer:
- Train accuracy = 99.51%
- Cross validation accuracy = 92.35%
- Test accuracy = 91.53%
- Test recall = 88.30%
- Test precision = 92.84%
- Test f1 score = 90.52%



GloVe + dropout

- Using no dropout regularization and GloVe 100d embeddings layer:
- Train accuracy = 99.71%
- Cross validation accuracy = 92.12%
- Test accuracy = 91.65%
- Test recall = 91.84%
- Test precision = 90.10%
- Test f1 score = 90.96%



no dropout + GloVe

BERT Model:

- The accuracies achieved are as follows:
- train accuracy = 98.47%
- validation accuracy = 97.31%

- We also see that predicting true for higher predicted probabilities yields better results:
- _predicting true for predicted probability >= 0.7:
- test accuracy = 97.734%
- recall = 97.91%
- precision = 97.20%
- f1_score = 97.55%

- _predicting true for predicted probability >= 0.5:
- test accuracy = 97.542%
- recall = 98.30%
- precision = 96.44%
- f1_score = 97.36%

- _predicting true for predicted probability >= 0.3: _
- test accuracy = 96.952%
- recall = 98.67%
- precision = 94.93%
- f1_score = 96.76%

Conclusion

The accuracies achieved by us are significantly higher than those achieved in the Stanford paper. The main reason for this can be the difference in the datasets used. The Stanford paper uses Reddit Data and Discussion Forum Data while we used News Headlines Data available on Kaggle. Reddit Data is quite unorganized compared to the News Headlines Data and needs to be pre-processed a lot.

BERT model was run on 30000 training examples and then on 55328 examples. We observed that when BERT was trained on 30000 examples, we achieved lower accuracy then the LSTM model but when we trained it on 55328 examples, BERT achieved higher accuracy. This shows that on smaller datasets, LSTM performs better but when we increase the size of the dataset, BERT performed better.

In the Future, Context in LSTM models can be included and also Hierarchical Attention Models can be implemented.

References:

- https://ai.googleblog.com/2018/11/open-sourcing-bert-state-of-art-pre.html
- https://www.analyticsvidhya.com/blog/2019/09/demystifying-bert-groundbreaking-nlp-framework/
- https://huggingface.co/bert-base-uncased#:~:text=BERT%20base%20model%20(uncased),difference%20between%20english%20and%20English.
- https://web.stanford.edu/class/archive/cs/cs224n/cs224n.1194/reports/custom/15791781.pdf

- https://machinelearningmastery.com/cnn-long-short-term-memory-networks/
- https://towardsdatascience.com/cnn-lstm-predicting-daily-hotel-cancellations-e1c75697f124
- https://www.analyticsvidhya.com/blog/2017/12/fundamentals-of-deep-learning-introduction-to-lstm/
- https://www.cs.toronto.edu/~rsalakhu/papers/srivastava14a.pdf
- https://www.researchgate.net/publication/285164623_An_Introduction_to_Convolutional_Neural_Networks
- https://keras.io/api/layers/pooling_layers/max_pooling2d/