Sarcasm Detection

Abdullah Irfan
a5irfan@uwaterloo.ca
University of Waterloo
Waterloo, Ontario

Nauman Ahmed
n78ahmed@uwaterloo.ca
University of Waterloo
Waterloo, Ontario

ABSTRACT

Sentiment analysis is often used in NLP to understand people's subjective opinions. However, the analysis results may be biased if people use sarcasm in their statements. Therefore, to correctly understand people's true intention, being able to detect sarcasm is crucial. In this project we propose to detect sarcasm in textual data. Sarcasm, especially, is key for sentiment analysis as it can completely flip the polarity of opinions. Understanding the ground truth, or the facts about a given event, allows for the detection of contradiction between the objective polarity of the event (usually negative) and its sarcastic characteristic by the author (usually positive), as in "I love the pain of breakup". Obtaining such knowledge is, however, very difficult. Previous efforts have been made to detect sarcasm based on sarcastic utterance in isolation. [4] introduced the use of both context (context sentence to which sarcastic reply is given) and reply text (sarcastic sentence) for detection which showed improvement in performance. The main models used in their paper were SVM and LSTM with attention. In our implementations however, we will investigate different attention models, along with introduction of fusion (early and late) on context and sarcastic reply text. We will investigate performance of models using LSTM, Bi-Directional LSTMs, BERT [7] and compare models using reply-only, and context and reply setups. BERT has never been used for sarcasm detection and we will show how a simple BERT classifier gives better results, and how BERT can be used for similar tasks.

1 INTRODUCTION

Sarcasm is an ironic utterance designed to cause pain with emotions such as mockery or insult. Sarcasm detection has many interesting areas of application such as marketing research and information categorization. Detecting sarcasm is a difficult task as its dependent on the context in which it was spoken, including the written tone. It can be expressed with the capitalization of words that are sarcastic, having a lot of nuances for effective detection.

Sarcasm detection could be performed in isolation to have the model just learn all the sarcastic instances. We however, investigated whether the context in which sarcasm occurred had an effect and whether adding attention to specific models improved our results.

In this project we explored the models explained by [4] and built upon his idea through various approaches, his idea was to incorporate context along with the sarcastic for sarcasm detection. The models we implemented included unidirectional simple LSTM, bi-directional LSTM, bi-directional LSTM with attention on replies (Early/ Late Fusion of Embedding's), Conditional LSTM variations. We paid close attention to hyper parameters such that the models would never overfit. Since we had a lot of models to explore we restricted our analysis with respect to Sarcasm Detection by only varying the size of data and number of epochs. We used Google's Word2Vec for our model embeddings.

We also implemented BERT for binary classification provided by [6], running the model in a simple setting by training on replies, and in a conditional setting where context and reply were both used. The BERT model takes a very long time to train, however, it performed better than all variations of LSTMs highlighted earlier. With minimal hyper-parameter tuning, BERT outperformed other methods, illustrating the benefits of using pre-trained models to obtain close to state-of-the-art results.

2 RELATED WORK

As stated previously, we wanted to expand on the work done in [4] which studied how the role of conversational context when merged with sarcastic comments behaves in various LSTM models. Prior to this paper, Ghosh et al. has explored sarcasm detection quite a bit, especially in his previous publication [5], which serves as a building block for his 2017 paper [4]. For this project we specifically focused on the latest paper as it has more deep learning implementations along with better results.

Furthermore, we investigated how attention should be incorporated for which we referenced [3] which talked about word level attention in neural networks to deal with long-term memory issues. We also got inspired by the multimodal fusion techniques discussed in [2] to combine the information in the context and reply texts to create new embeddings. For BERT we referenced [7] to understand limitations of fine-tuning if any, and then referenced [6] to understand the BERT implementation done in Pytorch.

3 APPROACH

In order to capture the true results, we will be discussing our baseline model setup and why our baseline was a solid starting point. We will then discuss the setups of various models we implemented.

Baseline Model

This project is inspired by [4], who thoroughly explored the role of conversational context and various LSTM models in sarcasm detection. For this class's final project, we intentionally chose his 2017 paper to establish a baseline as it introduces the concept of using sentiment and pragmatics, along with the idea to use replies and context together as a feature. All ideas besides pragmatics were used to establish the baseline.

For the features we utilized n-grams (unigram + bigrams), and lexicon-based features that capture the sentiment of the comment for which we used Vader Sentiment library in python.

We first developed a sentiment feature that gave us a score of whether the comment is positive or negative. We used the compound score to find out the overall sentiment of the sentence (we did this using VaderSentiment Analyzer). After that we made BOW features (unigrams + bigrams) with a maximum limit of 1000 features. We used the tf-idf vectorizer to get the BOW representation of our corpus. Concatenating with the sentiment scores we end up with 1001 features. We normalized the data and ran a simple Random Forest with n=100 trees which gave the results highlighted in Table 1.

The above steps were for reply only dataset, we then created 'context + reply'. For this we created the feature representation for the context data similar to what we did for the reply data and then concatenated horizontally such that we had a combined 2002 features. We then ran the Random Forest on it as well with n=100 trees giving the result in Result section as Context + Reply.

For the model evaluation we calculated Precision, Recall and F1, as well as Accuracy of our performance measurements along with the ROC curve. We did not remove punctuation nor did we lower case the data set because the context of sarcasm is better captured with it being inclusive.

As can be seen the results in Table 1 are not very satisfactory looking at the F1-scores and precision. As we progressed, to answer the question whether context helps, we established

Table 1: Baseline results with Reddit dataset

Dataset	Accuracy	F1	Precision	Recall
Replies	62.673	56.75	49.93	56.75
Context and Reply	68.182	52.23	41.06	48.49

better baseline models that were based on deep learning techniques.

Deep-learning Models

We applied 4 types of models; simple LSTM, bidirectional LSTM and BERT. We further implemented variations of unidirectional and bidirectional LSTMs. For BERT however, we used existing pytorch-pretrained BERT package used for binary classification and evaluations. The word embeddings used for the LSTMs were Google's Word2Vec. We will now discuss each model and its architecture.

Simple LSTMs.

- *Unidirectional LSTM with replies only:* We set up a unidirectional LSTM, which had no attention, and was trained only on the reply text corpus. The output of the last dense layer was binary classification of whether the sarcastic comment is sarcastic or not. Figure 1 illustrates the set up.

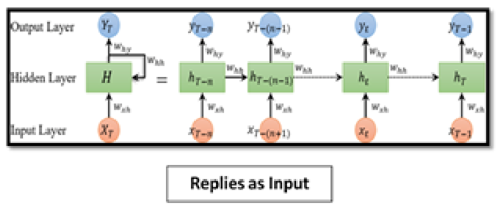

Figure 1: Unidirectional LSTM (Replies Only)

- *Conditional LSTM with context and replies:* The initial hidden state of Reply LSTM (Sarcastic Comment) is the last hidden state from the Context LSTM. This is the idea proposed by [4] where information is passed from the context LSTM to the succeeding reply LSTM. This is illustrated in Figure 2.

Bi-directional LSTMs.

- *Bi-directional LSTM with Reply only:* This LSTM would learn the forward sequence as well as the backward sequence to try and learn sentence polarity. From here on out this is why bi-directional LSTMs are used in our experiments.This is illustrated in Figure 3.

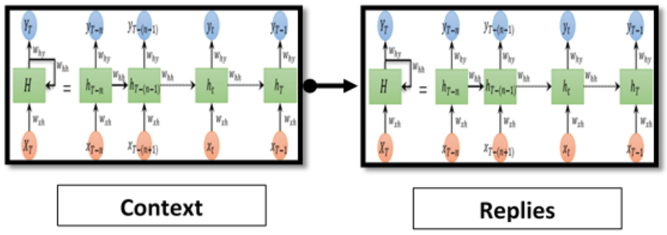

Figure 2: Conditional LSTM (Context and Replies)

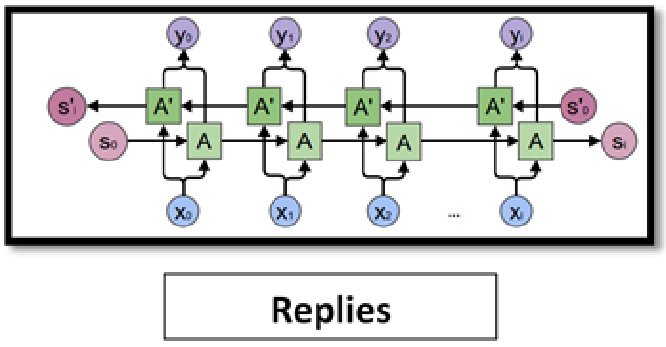

Figure 3: Bidirectional LSTM (Replies Only)

- *Bi-directional LSTM with Attention on Reply only:* This model gives different attention weights for each word during training. We have only used word level attention as this information helps understand which important word causes the sarcasm in the reply sentence. The model first applies Bi-directional LSTM generating hidden states. We then pass an attention layer to generate attention scores which summarizes the information. Applying attention to LSTM can solve some long term memory problems inspired by [3] implemented with the help of [9] and works as self-attention coined in transformers by [1]. Please note that this is not cross attention, where sentence level attention from context is passed to reply. We have illustrated this in an abstract manner in Figure 4.

- *Bi-directional Conditional LSTM at Context and Replies:* The model learns the context using Bi-directional LSTM, passes the forwaard and backward hidden and cell state weights to the reply Bi-directional LSTM. This is the conditional approach now implemented with bi-directional LSTM. This is illustrated in Figure 5

- *Bi-directional Conditional LSTM at Context and Replies, with Attention on Replies:* This is the conditional approach now implemented with bi-directional LSTM.

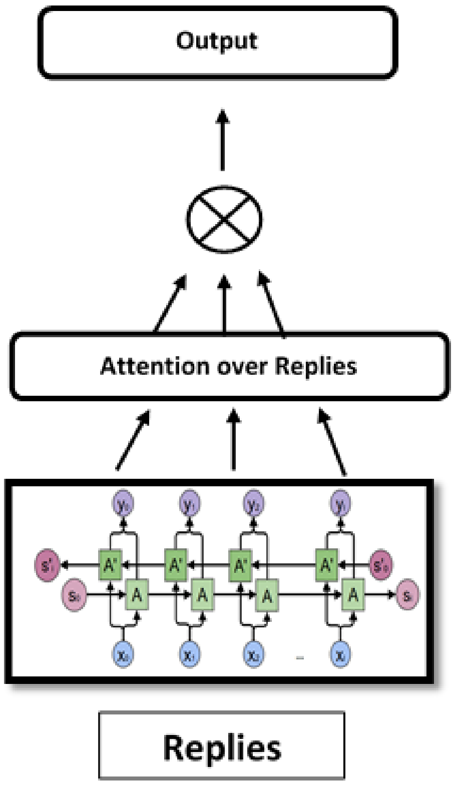

Figure 4: Bidirectional LSTM with Attention (Replies)

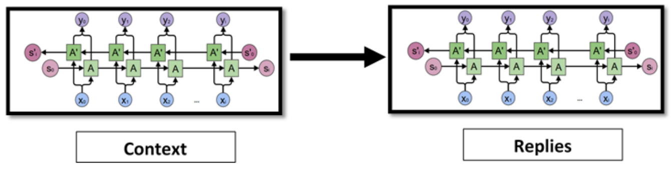

Figure 5: Bidirectional LSTM Conditional (Context and Replies)

This time we are adding importance of attention on replies side illustrated by Figure 6.

- *Bi-directional LSTM with Attention with Early Fusion of Embeddings:* We wanted to explore the effect of concatenating embeddings on the result. We defined early fusion as context and replies concatenated into one sentence. The embeddings are found for one long sentence, this embedding is passed to both the Bi-directional LSTMs, one which is for context and other Bi-LSTM for replies with attention. For the sake of simplicity we are providing an image of what early fusion is like in Figure 7.

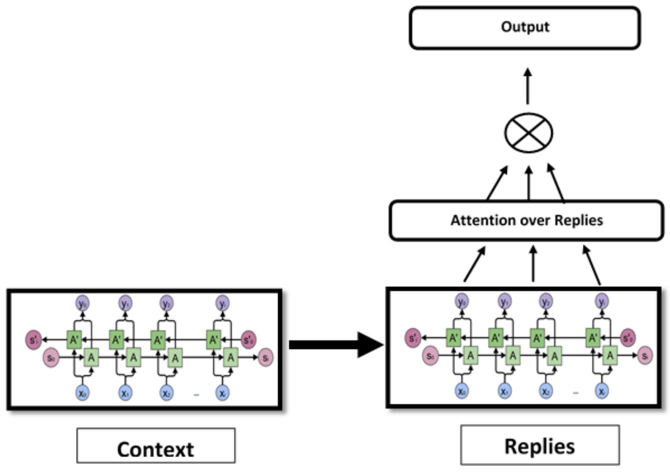

Figure 6: Bidirectional Conditional LSTM with Attention on (Replies)

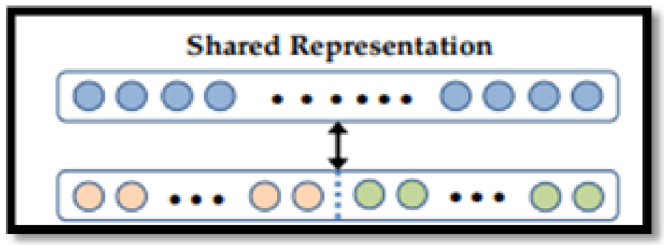

Figure 7: Early Fusion of Embedding with context and replies

- *Bi-directional LSTM with Attention with Late Fusion of Embeddings:* We defined Late fusion as context embeddings matrix made separately, reply embeddings made separately and then concatenated. This is illustrated in Figure [6]. So for instance if the dimensions were 300 for context and reply separately, then the merged matrix would have 600 dimensions. These combined embeddings formed are then passed as the first layer of both the BiLSTM (one for context and other for attention based BiLSTM for replies). This is illustrated in Figure 8.

BERT.

- *BERT on Replies Only:* The pre-trained model provided by [6] was utilized and then fine-tuned for our classification task with the help of [8]. The Reddit data was pre-processed to meet BERT's requirement which was to split into a train and test set. In the next step the data was tokenized and converted to input numerical features to be fed to the BERT model. The input sequence length was set to 78 which allowed to accommodate the majority of the input text tokens while truncating those that were very long. We used the 'bert-base-cased' pretrained tokenizer for this task as we wanted to retain the upper-cased words in our corpus.

  Next we fine-tuned the pretrained model for our classification task over 3 epochs. Once the model was fine-tuned its performance was measured over the test set we had created earlier. Refer to Figure 9 for an illustration of BERT.

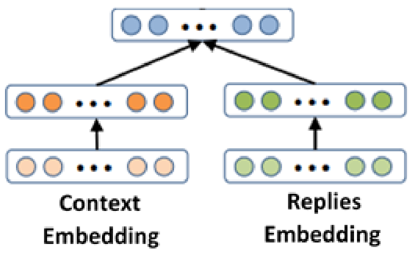

Figure 8: Late Fusion with separate embedding's concatenated

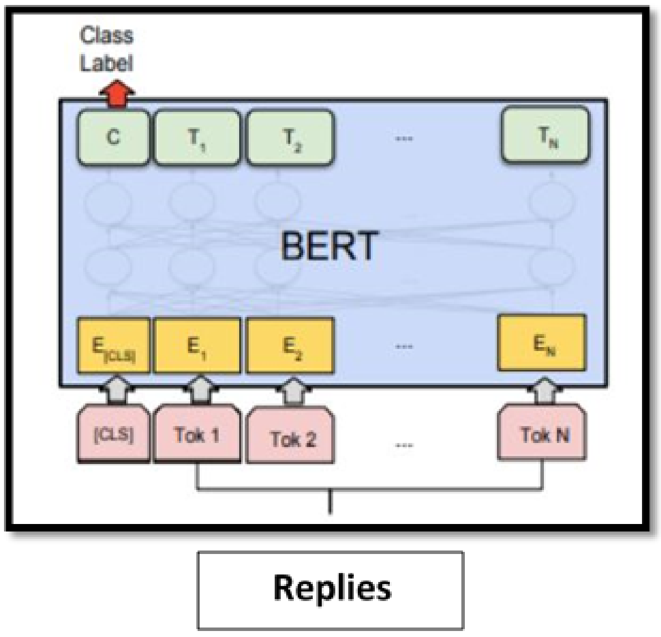

Figure 9: BERT applied on replies only

- *BERT Conditional on context and replies:* The steps here were quite similar to those we discussed earlier except that we input two sentences to the BERT model; the context and the reply. Each text was limited to a maximum sequence length of 78 as before. This allowed the BERT model to learn the relation between the sentences and then output the class label. See Figure 10.

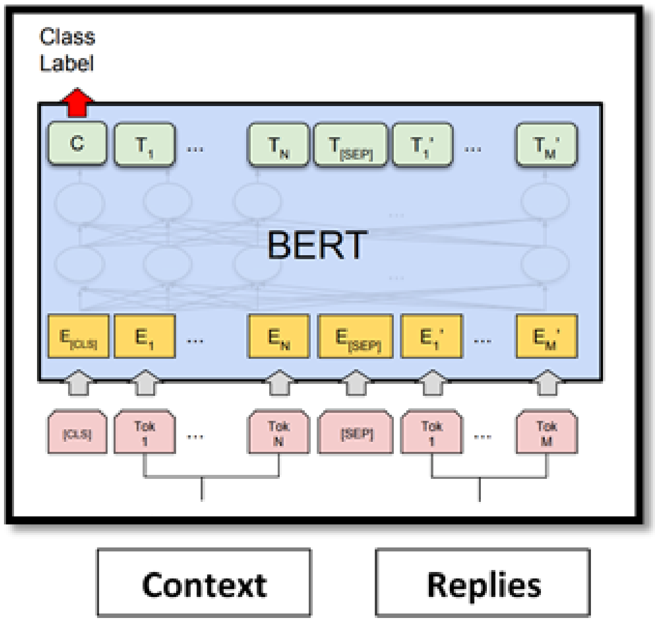

Figure 10: BERT applied on context and reply

4 EXPERIMENT SETUP

Dataset

The Dataset used for the project has been taken from Kaggle Reddit Sarcasm Detection available here. Data is a 1.3 million self-annotated corpus (balanced). The dataset contains context texts, response texts and label. For this project, we used 210k rows because of limited processing time. The data had other columns but for our experiments we used: parent_comment, comment and label. The parent_comment is the context statement, and comment column is the sarcastic reply, and the label defines whether the reply is sarcastic or not in binary.

- Training Data: All three columns identified and contained 132,300 rows.
- Validation Data: All three columns identified before and contains 56,700 rows.
- Test Data: Contained the three columns and 21,000 rows.

Model Configuration and Performance Tuning

For all the models we made, we kept the same configurations to ensure consistency. Furthermore, for performance tuning due to time constraint we just tuned on number of epochs. As we move from 20 epochs as seen in Figure 11, to 11 epochs as seen in Figure 12, we cater to over-fitting.

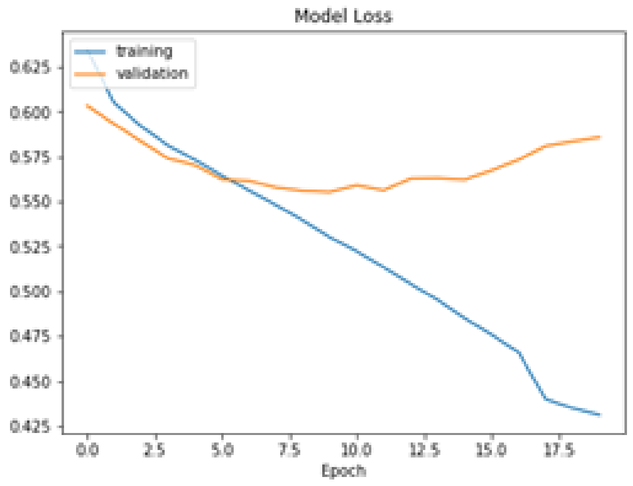

Figure 11: Conditional LSTM with 20 epochs

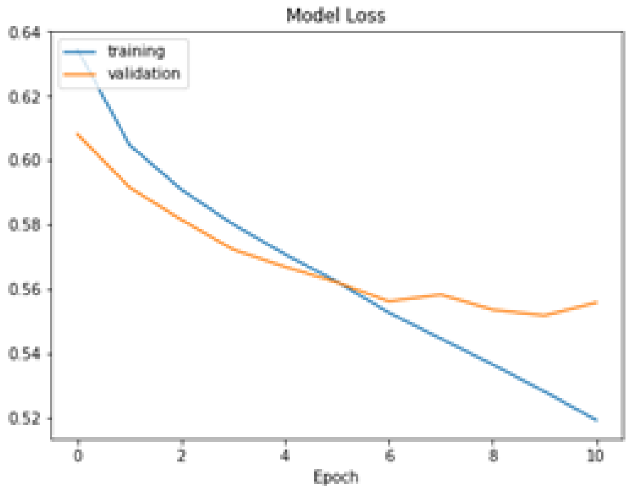

Figure 12: Conditional LSTM with 11 epochs

We also enabled early stopping to stop overfitting and used Keras ReduceLROnPlateau to adjust learning rates accordingly. Table 3 showcases the final configurations for the models made.

Model Evaluation Setting

Our models used an output dense layer with sigmoid activation that gave a prediction probability between 0 and 1. To optimize the model accordingly Adam Optimizer is used which sets the weight of each feature according to the output

Table 2: Summary of all the discussed models

Utterance Consideration	Model Detail
Replies Only	Baseline using Random Forest
Replies + Context	Baseline using Random Forest
Replies Only	Unidirectional LSTM
Replies Only	Bi-directional LSTM
Replies Only	Bi-directional LSTM with Attention
Replies Only	Bi-directional LSTM with Attention on Replies (Early Fusion of Embeddings)
Replies Only	Bi-directional LSTM with Attention on Replies (Late Fusion of Embeddings)
Replies + Context	Conditional LSTM
Replies + Context	Bidirectional Conditional LSTM
Replies + Context	Bidirectional Conditional LSTM (Attention on Replies)
Replies Only	BERT
Replies + Context	Conditional BERT

Table 3: Best Parameter Settings for the models

Parameter	Value
Learning Rate	0.001
Epochs	11
Per Epoch size	132297
Batch size	512
Embedding size	300
Hidden neurons size	128
Output size	1
BERT: max_seq_length	78
BERT: train_batch_size	24
BERT: number of Epochs	2
BERT: Learning Rate	2e-5

predicted.

For evaluation we used Precision, Recall, F1-Score and Accuracy for the binary predictions of sarcastic (1) / not-sarcastic (0).

5 RESULTS AND ANALYSIS

Table 4 lists all the results along with the model type and utterance consideration.

The results present a clear trend. Firstly, we find that bi-directional LSTM models perform significantly better in our application. This is because these models are able to look at both the past and future and therefore are able to understand the text context much better which is crucial in our application. In other words keep track of sentence polarity.

Secondly, we see that all variations of the conditional LSTM performs much better since the second LSTM initializes itself with the last hidden state of the first LSTM, which holds the context behind sarcasm, and therefore is able to process the reply text better to detect sarcasm, which we wanted to prove. This gives the model the ability to better understand the background in which the reply was delivered and therefore better categorize it as sarcastic or not-sarcastic.

Thirdly, adding word level self attention boosts the detection accuracy as it helps the model enhance its long term memory. We also experimented with various embedding fusions of reply and context text and find that the late fusion performs better which is intuitive.

Lastly, we find that BERT gives us the best results and that too when the context text is also input along with the reply text. BERT gives best results considering it is truly bidirectional and is able to understand the language relationships much better. Not to mention its large corpus aids with having weights assigned to every normally used words.

6 CONCLUSION

In this project we explored multiple deep learning techniques to classify text as sarcastic or not-sarcastic. We mainly focused on different variations of LSTMs, we trained BERT with response-only and response-and-context data derived from the 'Sarcasm on Reddit' dataset available on Kaggle. Our results clearly show that incorporating context information for sarcasm detection leads to better results. This holds true for most LSTM model implementations, and holds true for BERT as well. Furthermore, we show that applying word level self-attention helps the model to improve its long-term memory and gives us better results. Lastly, we show that a

Table 4: Experiment results

Utterance Consideration	Model	Precision	Recall	F1	Accuracy
Reply	Baseline	49.93	56.75	56.75	62.67
Reply + Context	Baseline	41.06	48.49	52.22	68.18
Reply	Unidirectional LSTM	71.33	57.51	63.68	71.14
Reply	Bidirectional LSTM	74.78	52.05	61.38	71.18
Reply	Bidirectional LSTM + Attention	69.7	59.61	64.26	70.83
Reply	Bidirectional LSTM + Attention(Early Fusion)	69.77	47.61	56.6	67.88
Reply	Bidirectional LSTM + Attention(Late Fusion)	69.78	59.47	64.21	70.84
Reply + Context	Conditional LSTM	68.98	61.59	65.07	70.91
Reply + Context	Bidirectional Conditional LSTM	72.22	56.68	63.52	71.35
Reply + Context	Bidirectional Conditional LSTM (Attention on Replies)	62.68	68.35	65.39	70.81
Reply	BERT	73.02	70.42	71.69	75.59
Reply + Context	BERT	74.21	70.73	72.43	76.36

pre-trained BERT classifier can achieve excellent results for sarcasm detection after it has been fine-tuned. This opens up another use case for BERT to achieve state-of-the-art results. We achieved an F1 score of 72.4% when the BERT was applied on the reply-and-context data. The results are expected to improve if a larger subset of data is used in exchange for longer training time. The improvement between LSTM (unidirectional, bidirectional and conditional) in comparison to BERT is significant. BERT improves the Average F1-score by 9% and average accuracy by 5.3%.

This project has given us a good understanding of the relationship between context and reply in sarcastic text. As a future goal we can choose to leverage this knowledge for possibly generating sarcastic replies given some context data. Other future works to improve performance would be to leverage user profiling into sarcasm detection. One area that we also wanted to explore is the effect of sarcastic word embeddings and not using Google's word2vec ones. One primary reason is that there are certain words we noticed that we believe would be overlooked by the Google Word2vec. Examples: 'OH DAYUUUMM, NoOOo WAAIII'. Presence of such words into our vocabulary could possibly improve the accuracy.

7 ACKNOWLEDGEMENTS

We would like to thank our professor Olga Vechtomova and TA Amirpasha Ghabus for guiding us throughout the course. We would also like to give special acknowledgement to Gaurav Sahu for providing us insight into techniques for multimodal learning (early and late fusion of embeddings).

REFERENCES

[1] Ashish Vaswani, Noam Shazeer, N. P. J. U. L. J. A. N. G. L. K. Attention is all you need. https://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf, 2017.

[2] Chi Thang Duong, Remi Lebret, K. A. Multimodal classification for analyzing social media. https://arxiv.org/pdf/1512.08756.pdf, 2016.

[3] Colin Raffel, D. P. W. E. Feed-forward networks with attention can solve some long-term memory problems. https://arxiv.org/pdf/1512.08756.pdf, 2016.

[4] Debanjan Ghosh, Alexander Richard Fabbri, S. M. The role of conversation context for sarcasm detection in online interactions. https://arxiv.org/pdf/1707.06226.pdf, 2008.

[5] Debanjan Ghosh, Weiwei Guo, S. M. Sarcastic or not: Word embeddings to predict the literal or sarcastic meaning of words. https://www.aclweb.org/anthology/D15-1116, 2015.

[6] huggingface. Pytorch-transformers. https://github.com/huggingface/pytorch-transformers, 2019.

[7] Jacob Devlin, Ming-Wei Chang, K. L. K. T. Bert: Pre-training of deep bidirectional transformers for language understanding. https://arxiv.org/pdf/1810.04805.pdf, 2019.

[8] Rajapakse, T. A simple guide on using bert for binary text classification. https://medium.com/swlh/a-simple-guide-on-using-bert-for-text-classification-bbf041ac8d04, 2019.

[9] takuoko. Bidirectional lstm and attention. https://www.kaggle.com/takuok/bidirectional-lstm-and-attention-lb-0-043, 2018.