Comparative Analysis of Sarcasm Detection in Reddit Comments

Kovidnath Pyla Reddy
Arizona State University,
kpyla@asu.edu

Viralam Sreedar Tharun Kumar
Arizona State University,
tviralam@asu.edu

Vishal Tyagi
Arizona State University,
vtyagi7@asu.edu

Anh The Nguyen
Arizona State University,
atnguy37@asu.edu

Sambhav Jain
Arizona State University,
sjain106@asu.edu

Abstract

The project uses different Natural Language Processing techniques to perform sarcasm detection on the Self Annotated Reddit Corpus. The main aim of the project is to perform a comparative analysis on the different approaches for detecting sarcasm. Since Sarcasm detection is a unique form of sentimental analysis we try to apply different pre-processing techniques in hopes of improving the results. Seeing as the Reddit dataset is also unique in its format we try to leverage the parent comments in hopes of understanding the context for better accuracy in classification. Attention mechanism and Bi-directional mechanisms have been utilized to incorporate the context into the responses in hopes of getting better results.

1 Introduction

Sarcasm detection has always been a topic of much interest in the field of Natural Language Processing. Especially because, the use of sarcasm is often a conversational tool that intends to convey the opposite of the literal meaning of the sentence with the help of irony. Previously Sarcasm was a tool almost exclusively used in verbal communication because of the need for vocal connotations to specific words in the sentence, to help the hearer understand the irony. But since the advent of social media sites and increase in textual communication, the use of sarcasm in natural language has become more common. The use of sarcasm in written form is generally hard to detect, even for human beings because of the various ways in which it can be applied. A lot of research in sarcasm detection during the past dealt with trying to build a vocabulary of words that could indicate that a sentence is sarcastic. But since then a lot of research has gone into trying to integrate the context of the sentence into the detection process. This project intents to analyze how various machine learning models perform on the Self Annotated Reddit Corpus[1] and to understand the importance of context in the detection of sarcasm.

There has been a lot of research done in the use of Convolutional Neural Networks (CNN) along with SVM on the SARC dataset (CASCADE [2]). Although there has been a lot of success in the CASCADE model, the context has been modelled by trying to understand the personality traits of the user by analyzing their previous comment patterns. The research paper, *Contextual bidirectional long short-term memory recurrent neural network language models: A generative approach to sentiment analysis*[3] by Debajan Ghosh et al uses a different approach by leveraging the previous comments in the thread and applying it to an LSTM model. The dataset that the LSTM was modelled was text data from different micro blogging websites, which provided a structured context in the form of parent text to the responses.

2 Previous Work

There has been a lot of previous research work done in the field of Natural Language processing in estimating the sentiment of sentences. We have modelled our research paper by following the work done in Textual Entailment by Chengjie Sun [14], the paper mainly deals with understanding how the Bi-directional LSTM model can help in estimating if a sentence has positive, negative or neutral entailment towards another sentence. We have utilized a similar methodology in solving our problem by attempting to classify sentences as sarcastic or non-sarcastic based on whether the response sentence entails the parent sentence, if it has positive or neutral entailment we classify the sentence as non-sarcastic where as if the response sentence contradicts the parent sentence then we classify it as sarcastic.

The research paper Anandkumar D. Dave[16] is a comprehensive study of various classification techniques for sarcasm detection. This research paper served as a good baseline to understand how each technique would perform for sarcasm detection, but since the paper deals mainly with the language of Hindi which has absolutely different ways in which sarcasm is conveyed we could not rely heavily on the results provided there for the different classifiers. But since the underlying mechanism for detecting Sarcasm was similar, we felt this paper was a huge contributing factor to our overall research process.

The research paper by Alex Kolchinski [15] used LSTM models to classify the sarcastic sentences, but it didn't explore the options of attention mechanism which was an added feature in our LSTM models, they have also tested the dataset on the CNN based CASCADE model, but this involved them using user statistics to come up with added weights to the model. We chose to stray away from this approach because each user had very few comments in the SARC dataset and it was not enough to actually come up with a reliable weight for the users. We instead chose to model heavily on the context using the attention mechanism to get better results.

3 Dataset

	Balanced		Unbalanced	
	Training	Test	Training	Test
Sarcastic	128541	32333	179700	44636
Non-Sarcastic	128541	32333	6575372	1637540

Table 1: Number of sarcastic and non-sarcastic comments in the dataset

The dataset we use to detect sarcasm is Self-Annotated Reddit Corpus (SARC), a large corpus for sarcasm research and for training and evaluating systems for sarcasm detection[1]. It is divided into two part (balanced and unbalanced) and demonstrated by the table above:

Reddit comments from December 2005 have been made available due to web-scraping 4 ; we construct our dataset as a subset of comments from January 2009-April 2017, comprising the vast majority of comments and excluding noisy data from earlier years.

Training or testing dataset is a CSV format and every row data in the file contains three parts:

1. Parents: the comments begin for the conversation. This part includes one or more comments.
2. Response: these are responses to the parent thread.
3. Label: each label assigned to each pair of every response and last comments in the file, every part is separated by separator(|).

For example, one of the records can be taken as :

635m1n dfrqnbp|dfshqbg dfrz30u|1 0

In the above example, *635m1n* and *dfrqnbp* are the ID of parents comments. *dfshqbg* is the response ID of *dfrqnbp* and *dfrz30u* is also the response ID of *dfrqnbp*. 1 means (*dfrqnbp*, *dfshqbg*) sarcastic and 0 means (*dfrqnbp*, *dfrz30u*) non-sarcastic.

4 Methodology

After the data preprocessing, the word vectors will be separately processed for each of the algorithms mentioned below, the models for each of them will be trained on different structures of the same dataset and the results will be plotted by running on the saved models

- Logistic Regression
- Word2vec
- LSTM
- Attention based LSTM
- Bi Directional RNN
- SVM

4.1 Data Preprocessing

There were three main files which had to be pre-processed, the balanced training and balanced testing files were encoded as unique keys instead of the actual comments. To get the sentences we had to load another JSON file which had the key-value pairs for each of the sentences. After the words were fetched they were further preprocessed to make it optimum for the task of Sarcasm detection. The main NLP processes used were:

1. Stop word removal: Removing words from the sentence that do not contribute to the meaning of the sentence, these words are predefined as part of the NLTK stop word corpus.
2. Lemmatization: This process involves grouping together adjacent words to form a more common phrase or making a shortened form of the word into its recognizable form.
3. Unwanted symbol removing: many regular expressions have been written to filter any unwanted symbols from the sentences
4. Tokenization: Splitting the words into smaller segments called tokens to make it easier while feeding into the word2vec model

4.2 Logistic Regression

Logistic Regression is a statistic method for analyzing a dataset in which there are one or more independent variables that determine an outcome. The outcome is measured with a dichotomous variable, in our case classification of the comment as either sarcastic or non-sarcastic.

The goal of this algorithm is to find the best fitting model to describe the relationship between sarcasm and a set of independent variables. Logistic regression uses Maximum likelihood estimation to find a set of regression coefficients that maximizes the probability of observing the data given in the training set.

4.3 Word2vec

Most of the previous work in sarcasm involved using pre-trained word embeddings such as Glove vectors [4] because the sarcasm detection didn't involve using word embeddings based on the context of the sentences. Although using pre-trained word embeddings provided good results we wanted to use a CBOW (Common Bag of words) model to train our own Word2Vec model based on the SARC dataset[ref1]. The idea behind training the Word2Vec model is that the meaning of the word is dependent on the company it keeps, which is a very conducive training technique for our dataset which heavily depends on the context and accompanying words.

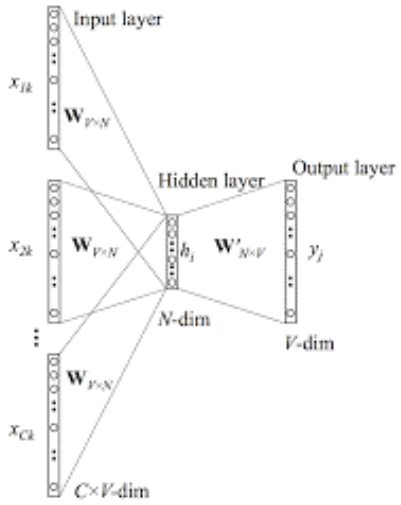


Figure 1: Word2Vec Model Training

Our Word2Vec model is trained is by using a Neural network approach. The context of each word is taken as an input and the vector representation of the word is predicted using this. The model has three layers: Input layer, Hidden layer, Output layer. Each of the V words from the context are fed as one hot vectors to the input layer, the hidden layer consists of N neurons and the output is a V length vector with the elements being the SoftMax values. We must train a weight matrix which helps map the input to the hidden layer $h = x^T W$. There is also another weight matrix which maps the hidden layer to the output layer.

4.4 LSTM

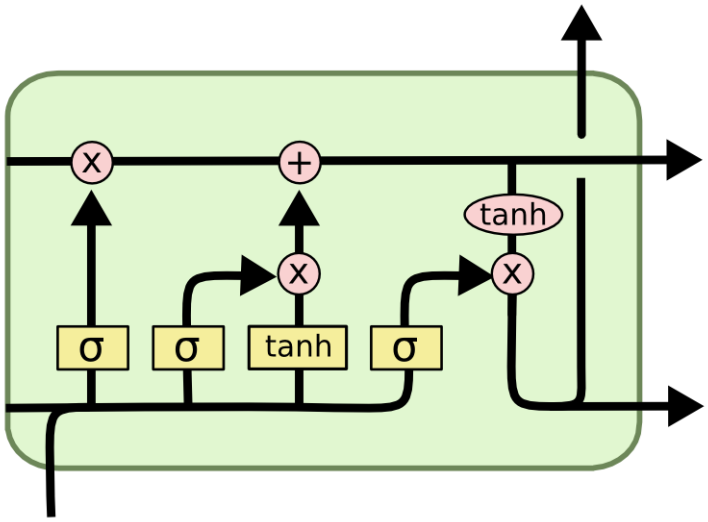


Figure 2: LSTM Architecture

The LSTM (Long short-term memory) is a type of recurrent neural network, which allows for retention of information from long distances from the word for which the embedding must be found.

Although Recurrent neural networks are basically neural networks with loops which help the retention of information about words which are further apart will be forgotten. This is avoided in the case of an LSTM cell because of the existence of the forget and remember gates. With the use of these gates the LSTM cell learns what information is valuable and what information is expendable when it comes to the label prediction. The LSTM model is extremely useful as compared to common bag of words approach because:

1. The balanced reddit dataset is built in such a way that each parent has exactly one sarcastic comment and one non-sarcastic comment, so the bag of words approach would fail when it comes to using a CBOW model.
2. When an LSTM model is used different parts of the parent sentence will be remember for each of the sarcastic and non-sarcastic comment.
3. But since the same parent is present for both the sarcastic and non-sarcastic comment there would be a lot of false positives.

The way that the remember and forget gates operate such that this model can work is as follows:

$$S_j = R_{LSTM}(S_{(j-1)}, x_j) = [c_j; h_j]$$

$$c_j = c_{(j-1)}.f + g.i$$

$$h_j = tanh(c_j).o$$

$$i = \sigma(x_j W^{xi} + h_{(j-1)} W^{hi})$$

$$f = \sigma(x_j W^{xf} + h_{(j-1)} W^{hf})$$

$$o = \sigma(x_j W^{xo} + h_{(j-1)} W^{ho})$$

$$g = tanh(x_j W^{xg} + h_{(j-1)} W^{hg})$$

$$y_j = O_{LSTM}(s_j) = h_j$$

4.5 Attention Based LSTM

To improve on the LSTM neural network, we chose to add an attention layer to make the precision of what words the neural network concentrates on. Although the LSTM model helps the neural network remember the previous works in the sequence, the attention layer helps refine which words contribute the most to the overall result. The major advantage that the attention mechanism provides over the Vanilla LSTM model is that it helps the network from forgetting words that are further away in the sentence. This is especially important in our sentence structure because the context and responses are a maximum of 30 words each, and for the effects of the context to be considered significantly during our testing phase we must make the network remember words which are quite distant. Previously the attention mechanism has been largely used for machine translation tasks to provide context, but the Attention based LSTM has been evidenced to work on specific sentimental tasks such as Aspect-level sentimental analysis as per the research done by Yequan Wangand Minlie Huang [5]. Since context is incredibly important for the task of sarcasm detection as well, we have chosen to leverage the mechanism here.

4.6 Bidirectional RNN

RNN only tells us about the context of left-right sentence. Now we want to see the context in the other side of sentence by looking at it from right to left to understand whole context of the sentence. Therefore, a Bidirectional LSTM can help us achieve this goal.
A Bi Directional LSTM processes input sequences in both directions with two sub-layers in order to account for the full input context. These two sub-layers compute forward and backward hidden sequences $\vec{h}$ and $\overleftarrow{h}$ respectively, which are then combined to compute the output sequence y.

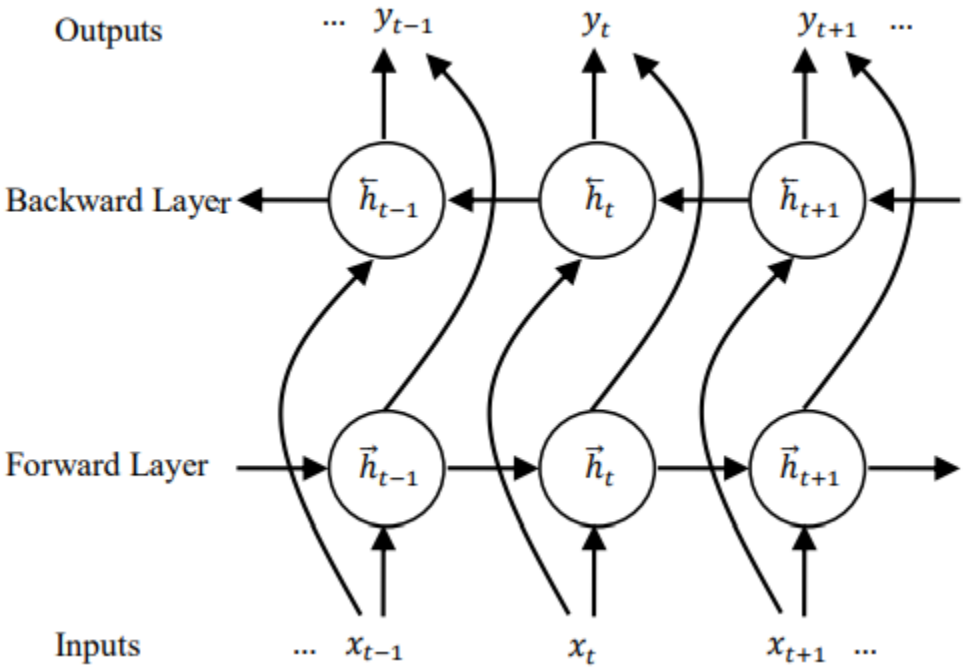


Figure 3: Architecture of a Bi Directional LSTM, every output depends on the whole input sequence

4.7 Support Vector Machines

A machine learning algorithm based on the feature vectors generated from the data and Word2Vec was used to train a classifier. The classification algorithm used is support vector machine (SVM) due to its simplicity and effectiveness in binary classification. For one class SVM, we want to solve the following optimization problem:

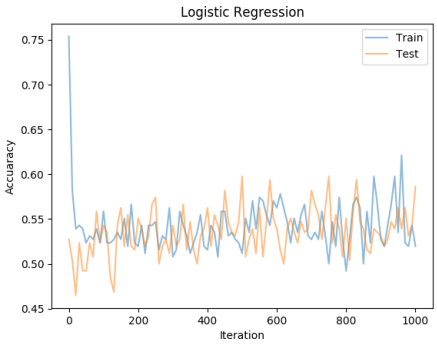
$$\sum_{n=1}^{N} \xi_n + \frac{1}{2} \mid w \mid^2$$

where the parameter C $>$ 0 controls the trade-off between the slack variable penalty and the margin.

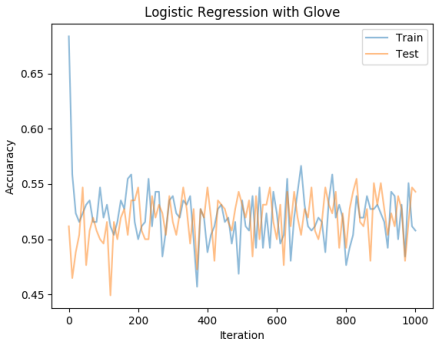
5 Results

The raw unbalanced dataset consists of very disproportionate comments which causes the network to overfit. The accuracy results that we have gotten from the unbalanced data set are actually inflated and to test this hypothesis we tested the model created from the unbalanced dataset on the balanced test data, which in turn provided us an accuracy of 52.24%. Thus we chosen the balanced dataset as our principal dataset for all our further calculations and deliberations.

Sarcasm detection has been performed on 5 distinct algorithms namely – LSTM, Bidirectional LSTM, LSTM with attention mechanism, SVM and Logistic regression. As we can see from the results provided in the tables above that all three LSTM modelsprovide higher accuracy results as compared to the simpler SVM and logistic regression models. This is mainly because the Long Short-Term Memory model helpsus better understand the context behind the comments, as opposed to building a normal vocabulary out of the sarcastic sentences. But the differences between the LSTM models itself can only be understood when we dig deep into the precision and recall scores. All the algorithms were implemented in some specific hyperparameters which give us the best performance (Learning Rate: 0.01, Batch Size: 256, 100 Dimensional Vector for Glove and our trained Word2Vec, except for only SVM with 0.1 Learning Rate)

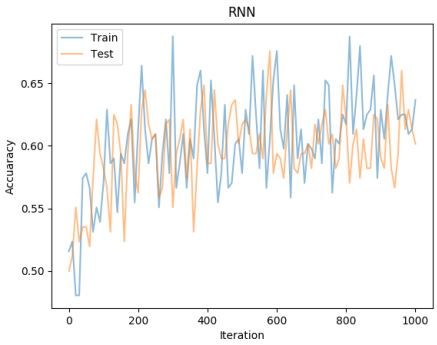


(a) Accuracy Plot of Logistic Regression

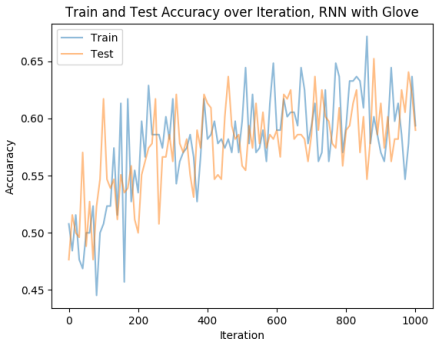


(b) Accuracy Plot of Logistic Regression with Glove

Figure 4: Logistic Regression Accuracies

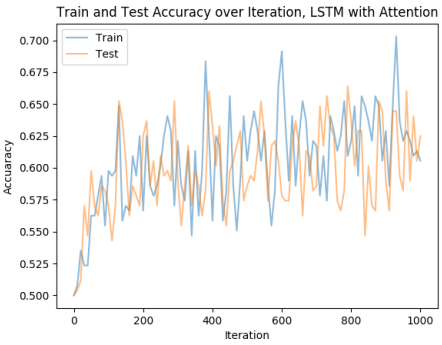


(a) Accuracy Plot of LSTM

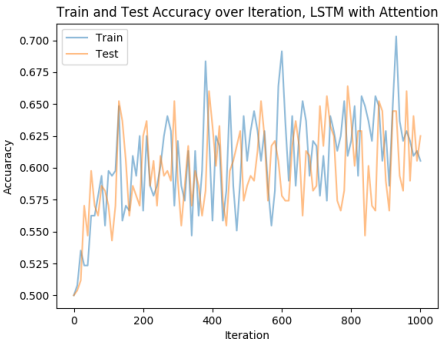


(b) Accuracy Plot of LSTM with Glove

Figure 5: LSTM Accuracies

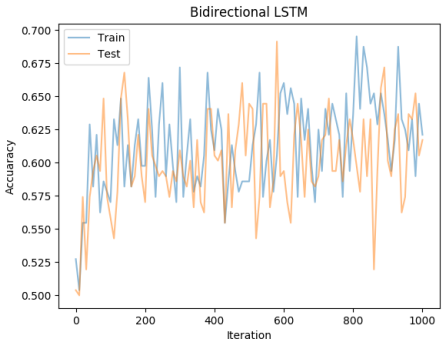


(a) Accuracy Plot of Attention LSTM

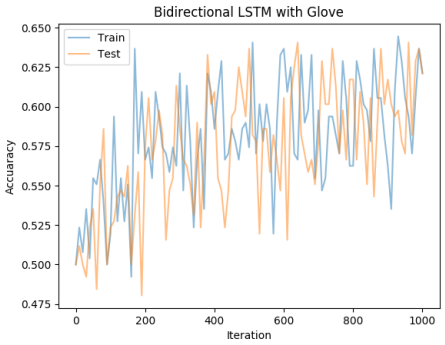


(b) Accuracy Plot of Attention LSTM with Glove

Figure 6: Attention Based LSTM Accuracies

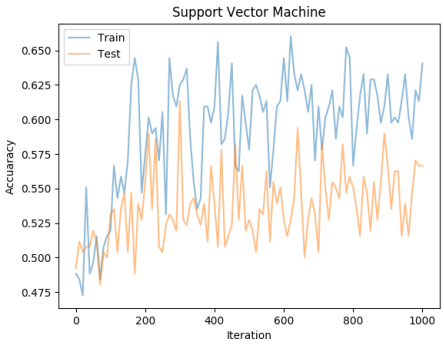


(a) Accuracy Plot of Bi-Directional LSTM

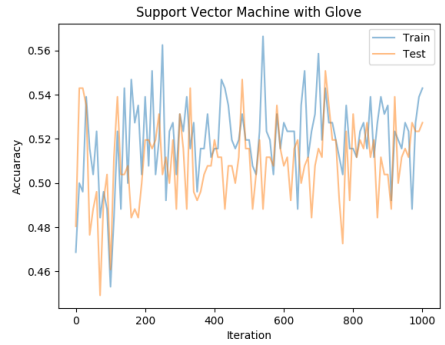


(b) Accuracy Plot of Bi-Directional LSTM with Glove

Figure 7: Bi-Directional RNN Accuracies

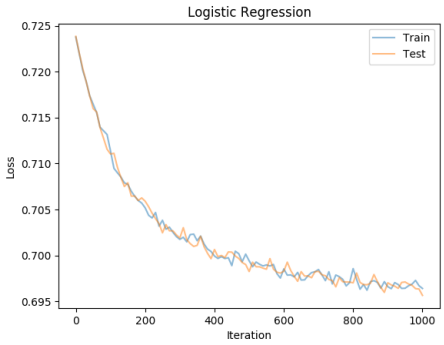


(a) Accuracy Plot of SVM

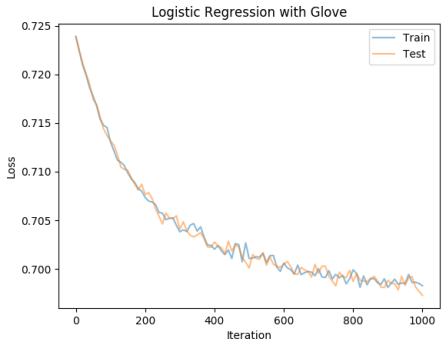


(b) Accuracy Plot of SVM with Glove

Figure 8: SVM Accuracies

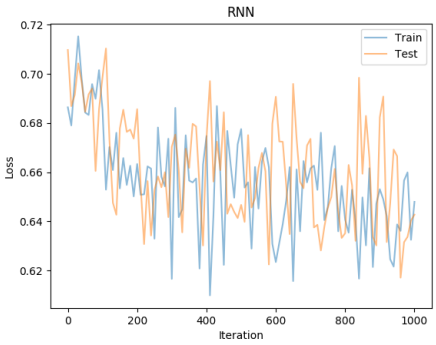


(a) Loss Plot of Logistic Regression

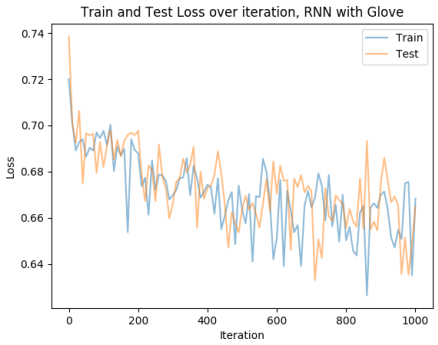


(b) Loss Plot of Logistic regression with Glove

Figure 9: Logistic Regression Loss



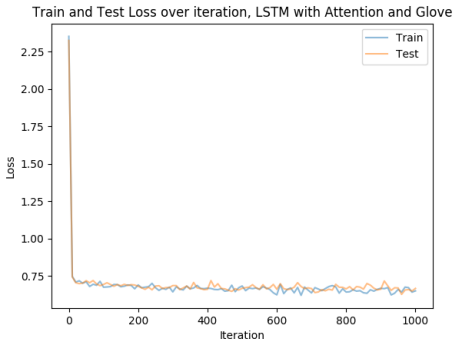
(a) Loss Plot of LSTM



(b) Loss Plot of LSTM with Glove

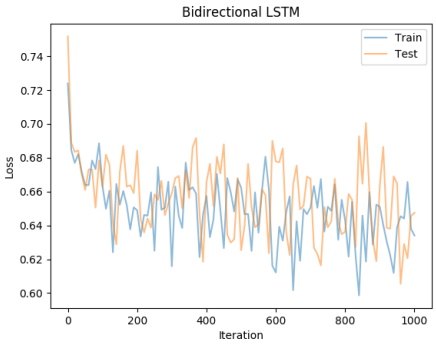
Figure 10: LSTM Loss

(a) Loss Plot of LSTM Attention

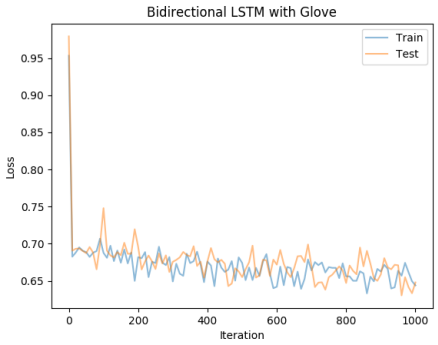


(b) Loss Plot of LSTM Attention with Glove

Figure 11: Attention Based LSTM Loss

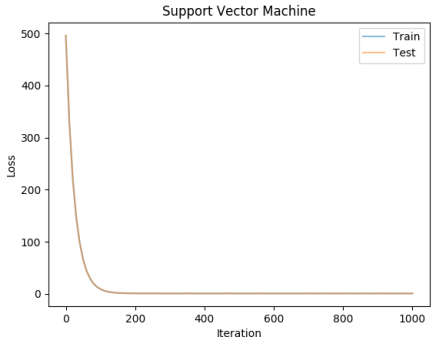


(a) Loss Plot of Bi-Directional LSTM

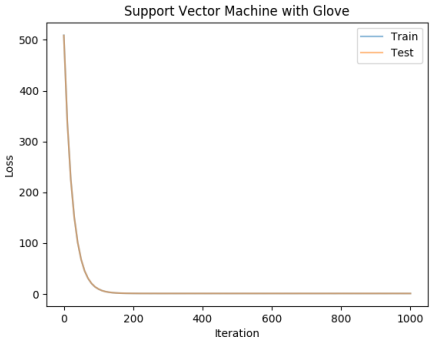


(b) Loss Plot of Bi-Directional LSTM with Glove

Figure 12: Bi-Directional RNN Loss



(a) Loss Plot of SVM



(b) Loss Plot of SVM with Glove

Figure 13: SVM Loss

	Accuracy	Confusion Matrix	Precision	Recall	F1-Score
Logistic Regression	0.5176	[8977 , 23279] [7840 , 24416]	0.5119	0.7569	0.6108
LSTM	0.5933	[20484 , 11772] [14523 , 17733]	0.6010	0.5497	0.5742
Attention Based LSTM	0.5981	[19802 , 12454] [13478 , 18778]	0.6012	0.5822	0.5915
Bi-Directional LSTM	0.5981	[18407 , 13849] [12077 , 20179]	0.5930	0.6256	0.6089
SVM	0.5063	[3293 , 28963] [2887 , 29369]	0.5035	0.9105	0.6484

Table 2: Accuracy with Glove Vectors

	Accuracy	Confusion Matrix	Precision	Recall	F1-Score
Logistic Regression	0.5418	[12930 , 19326] [10232 , 22024]	0.5326	0.6827	0.5984
LSTM	0.6174	[22016 , 10240] [14475 , 17781]	0.6345	0.5512	0.5899
Attention Based LSTM	0.6187	[21786 , 10470] [14132 , 18124]	0.6338	0.5619	0.5957
Bi-Directional LSTM	0.6150	[19884 , 12372] [12468 , 19788]	0.6153	0.6135	0.6144
SVM	0.5487	[17145 , 15111] [14003 , 18253]	0.5470	0.5659	0.5563

Table 3: Accuracy with Word2Vec Model

From the figures shown below we can get a detailed understanding of how each of the models were trained and tested in every iteration. We can conclude that the LSTM models were far superior in every iteration with respect to the accuracy and precision, but it gets interesting to analyze the differences between each of the LSTM models. We notice that although the Accuracy and Loss plots are almost the same in every iteration, the main difference can be gleaned from the table where we have mentioned the Precision and Recall values. From the tabular results we can conclude that although the attention model has provided similar accuracies as the other LSTM models, it trumps these models when it comes to the accuracy in detecting sarcasm in sentences.

6 Conclusion and Future work

After thorough analysis on the different models for sarcasm we have realized that the neural network models provide better results because of the integration with the context, so our initial assumption that the context is useful for the sarcasm detection is correct. Simpler models such as SVM and Logistic regression tend to under-perform because it does not take in consideration of context.

During the course of our project, our group tried to propose different approach in order to improve accuracy of Sarcasm Detection. Using user's traits is one of the important criteria to predict the probability of giving sarcastic comments from users. Understanding the history of a users comments is important to determining if the comment is sarcastic. Our team planned to utilize User Id and User comments from the dataset and aggregate all the information of user, such as how many sarcastic comments given by a specific user. So that we could calculate probability of predicting a sarcastic sentences from users based on their comments in the past. Naive Bayes would then be applied on this situation to solve the Sarcastic Detection problem. Unfortunately, The dataset does not include enough information of the users to allow us to analyze, there was a discrepancy in the amount of comments written by each user. In the future, we would like to collect more data from Reddit or other social media website, especially user information, to train new model by Naive Bayes and improve the accuracy.

References

[1] Khodak, Mikhail, Nikunj Saunshi, and Kiran Vodrahalli. "A large self-annotated corpus for sarcasm." arXiv preprint arXiv:1704.05579 (2017).

[2] The Bidirectional Language Model, URL: *https://medium.com/@plusepsilon/the-bidirectional-language-model-1f3961d1fb27*

[3] Mousa, Amr, and Björn Schuller. "Contextual bidirectional long short-term memory recurrent neural network language models: A generative approach to sentiment analysis." Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers. 2017.

[4] Pennington, Jeffrey, Richard Socher, and Christopher Manning. "Glove: Global vectors for word representation." Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). 2014.

[5] Kolchinski, Y. Alex, and Christopher Potts (2018) "Representing Social Media Users for Sarcasm Detection", *arXiv preprint arXiv:1808.08470*

[6] `https://www.kaggle.com/rmisra/news-headlines-dataset-for-sarcasm-detection`, "News Headlines Dataset For Sarcasm Detection."

[7] Joshi, Aditya, Vaibhav Tripathi, Kevin Patel, Pushpak Bhattacharyya, and Mark Carman. (2016) "Are word embedding-based features useful for sarcasm detection?", *arXiv preprint arXiv:1610.00883*

[8] Zhang, Meishan, Yue Zhang, and Guohong Fu. (2016) "Tweet sarcasm detection using deep neural network", *In Proceedings of COLING 2016, The 26th International Conference on Computational Linguistics: Technical Papers, pp. 2449-2460.*

[9] Liebrecht, C. C., F. A. Kunneman, and A. P. J. van Den Bosch. (2013) "The perfect solution for detecting sarcasm in tweets #not."

[10] Kuo, Po Chen, Fernando H. Calderon Alvarado, and Yi-Shin Chen (2018) "Facebook Reaction-Based Emotion Classifier as Cue for Sarcasm Detection", *arXiv preprint arXiv:1805.06510*

[11] Hazarika, Devamanyu, Soujanya Poria, Sruthi Gorantla, Erik Cambria, Roger Zimmermann, and Rada Mihalcea. (2018) "CASCADE: Contextual sarcasm detection in online discussion forums.", *arXiv preprint arXiv:1805.06413*

[12] G, Dr. Vadivu & Chandra Sekharan, Sindhu. (2018) "A COMPREHENSIVE STUDY ON SARCASM DETECTION TECHNIQUES IN SENTIMENT ANALYSIS", *International Journal of Pure and Applied Mathematics.*

[13] Joshi, Aditya, Pushpak Bhattacharyya, and Mark J. Carman. (2017) "Automatic sarcasm detection: A survey.", *ACM Computing Surveys (CSUR) 50.5 (2017): 73.*

[14] Yang Liu, Chengjie Sun, Lei Lin , and Xiaolong Wang. (May 2016) "Learning Natural Language Inference using Bidirectional LSTM model and Inner-Attention."

[15] Y. Alex Kolchinski , and Christopher Potts. "Representing Social Media Users for Sarcasm Detection."

[16] Anandkumar D. Dave and Nikita P. Desai. "A comprehensive study of classification techniques for sarcasm detection on textual data."