INTERNATIONAL JOURNAL OF ADVANCE RESEARCH, IDEAS AND INNOVATIONS IN TECHNOLOGY

ISSN: 2454-132X
Impact Factor: 6.078
(Volume 7, Issue 3 - V7I3-1753)
Available online at: https://www.ijariit.com

Sarcasm Detection in English and Hindi Sentences

Aakash Kumar
aakash.sain08@gmail.com
Maharaja Agrasen Institute of Technology, Delhi

Pashmeen Kaur
pashmeen2000@gmail.com
Maharaja Agrasen Institute of Technology, Delhi

M. L. Sharma
mlsharma@mait.ac.in
Maharaja Agrasen Institute of Technology, Delhi

K. C. Tripathi
kctripathi@mait.ac.in
Maharaja Agrasen Institute of Technology, Delhi

ABSTRACT

In the growing advancement of Natural Language Processing, sarcasm detection, a part of sentiment analysis is still a challenge. It is important to detect sarcasm in the statement, especially in open platforms where people are free to express themselves, to identify the correct intended meaning of the statement mentioned. Sarcasm is an important processing problem in the NLP field of sentiment analysis, as it serves as an interface between communicating humans and machines, also such sentences can change the polarity of a sentence, which might result in wrong sentiment analysis. Here, in this project, we going to work upon sarcasm detection in news headlines, which we can get online or in news articles. We made use of LSTM[1][4], an artificial recurrent neural network architecture used in the field of deep learning[8], built around our collected dataset. The model is trained over the pre-processed dataset, cleaned with the help of the nltk library enabling it to create word embeddings, remove stopwords, etc.

Keywords— *Sarcasm detection, sentiment analysis, NLP, Neural networks, RNN, LSTM, a detection system*

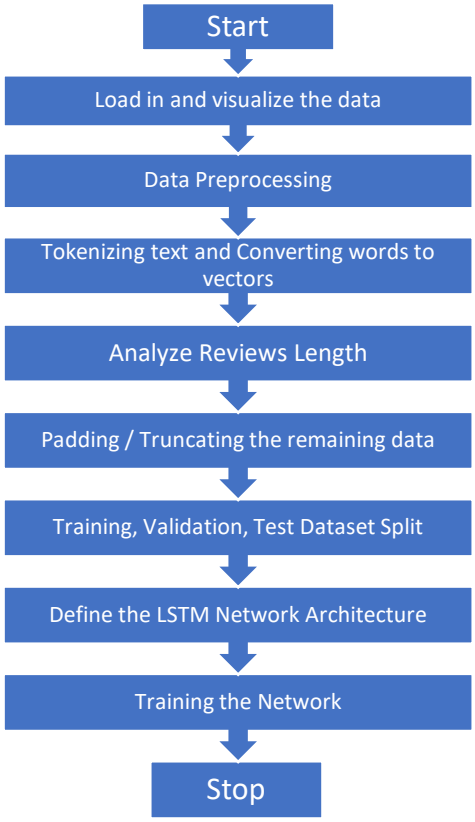
1. INTRODUCTION

Sarcasm is "a sharp, bitter, or cutting expression or remark". According to the oxford dictionary, "Sarcasm is the use of language that normally signifies the opposite to mock or convey contempt." People often uses positive words in a sentence to denote a negative message or vice-a-versa, which may vary according to the people's receptivity and understanding of the notion. Social Media such as Twitter, exhibit rich sarcasm phenomena, upon which many recent works have been performed and researched. But in our project, we are making use of news headlines and day-to-day statements to detect sarcasm in English and Hindi language respectively, to get a different set of results. Many text classification methods have been used to deal with this emerging subject of interest with the rise of volume of content being fabricated on social media platforms and a need to analyze it closely. In-text classification, sarcasm detection is considered to be an essential tool as it has many implications for several areas, including security, health, sales, etc. Companies can analyze customer's feelings, a person can know the true intention of the statement made by another, etc. with the help of sarcasm detection technique. This tool can provide crucial help for those companies to boost their product quality and people a better understanding. Sarcasm identification itself is an essential subtask in sentiment analysis, as it helps in extracting the implicit information the user tries to convey within his/her messages or views one share with others.

Being from India, a mass of people conversing in Hindi and English both, it will be a great idea to introduce a sarcasm detector for both languages. Therefore, in this project, we analyzed how the detection of sarcasm can be performed in both English as well as the Hindi language to bring diversity in this emerging field of Natural Language Processing (NLP). To do so, we make use of one of the Deep Learning algorithms, an artificial recurrent neural network, LSTM, which stands for Long-short-term memory. Here, in this paper, we are not building a model that will use a multilingual input, a mixture of two or more languages, instead, we are analyzing to see how well an LSTM model can detect sarcasm in different languages, how flexible and how well it can perform with a language other than Hindi. So, to do the same, we tracked the results for English input and Hindi input built on the different LSTM models accordingly and compared the performance of our LSTM algorithm with other deep learning algorithms like Bi-directional LSTM, Bi-directional with attention layer, and RNN.

2. FLOW OF WORK

Figure below demonstrates the work flow.



3. PROPOSED METHODOLOGY

3.1. Dataset

Here in this paper for the English language, we are making use of news headlines, collected across from various sources, and given labels 0 for non-sarcastic headlines and 1 for sarcastic headlines [3.a].

This dataset contains a total of 26709 headlines which is a bit unbalanced as it contains 14985 headlines that are non-sarcastic and 11724 headlines that are sarcastic.

For the Hindi language, we chose to select daily conversing dialogues used by people of India, which was collected from different social media platforms, news headlines, etc.
This dataset consists of 16179 Hindi sentences, in which 6051 are sarcastic and 10128 non-sarcastic, labelled as 1 and 0 respectively[3.b].

3.2. Data Pre-processing

Since the English dataset was collected from Kaggle, it doesn't require a lot of pre-processing, just removal of stopwords and any numbers, if present, was to be done. Also if there was any rare word, i.e., a word that has occurred less than 10 times we removed those words as well.

In the Hindi dataset, we approached similarly by removing stopwords from the sentences and least frequent words too before processing it in our models.

3.2.1 Converting words to tokens: Tokenization is the process of breaking down a text into words. Tokenization can happen on any character; however, the most common way of tokenization is to do it on space character. Using those separated words, we simply convert words to something which the computer can understand, i.e., numbers. It refers to giving each word in a sentence a unique number, which represents each word. We make data out of them, by converting sentences to a sequence of numbers. Each number represents a word in a sentence. This sequence of numbers, the data, we pass to a Neural Network to train and process it.

3.2.2 Word Embedding/Sequence of tokens: Word embedding visualizes words into a continuous low-dimensional depiction. By this kind of embedding related words converge closer to each other in vector space [5][6]. It is a sequence of integer values forming sentences as they were before when they were sentences with actual words. It is important to convert words into integers (or tokens) and then forming a sequence of those tokens to resemble actual sentences and preserve the contextual relation between words. As we need numbers to feed into our neural network[5][6], so to create a sequence of words all we need to do is to use the instance we created of tokenizer and convert texts to sequence with reviews as the argument and it will return a huge list containing small lists of a sequence of integer values.

3.3. Deep Learning Models

Here, we propose 4 different models to experiment with the data we have collected, which are RNN[2][7], LSTM[4][6][7], Bi-directional LSTM[9], and Multi-layered Bi-directional LSTM.

3.3.1 Recurrent Neural Network: To understand LSTM first we need to know about RNN[3][7]. It is already mentioned that LSTM is a special kind of RNN. When we read an essay or a paragraph, we read each word and take note of each word as we move forward. Every word in a paragraph is important in building a context, of what this paragraph is all about. Hence, we need to have an understanding of each previous word to understand the coming/next word(s). A simple Neural Network cannot do such a task. In a simple Neural Network, each state is independent of the other states in the network and no state has any influence on other states. RNNs were created to solve this problem with neural networks.

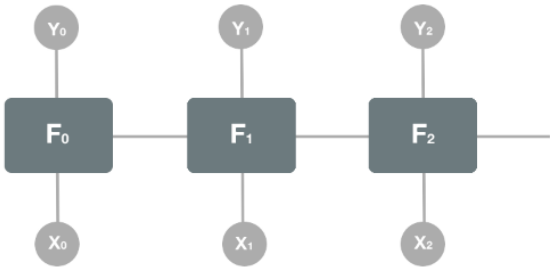


Fig. 3.3.1.1 Simple RNN architecture

Figure 1, a part of a neural network, shows us how RNN not only gives us an output at every state but also gives feed-forward to the next state to maintain context. Here, F_0 takes an input say, $X_{0,}$ and gives an output $Y_{0,}$ and passes on the information to the next step in the network. It has only one tanh layer forming a very simple repeating chain structure.

In our work, we created an RNN model with 5 layers: Embedding layer (16,8) shape, Simple RNN layer of 64 nodes, two dense layers consisting of 64 and 1 output node respectively, and a dropout layer in between of both with 50% dropout ratio.

3.3.2. LSTM Network: LSTMs have been successfully applied to binary text classification problems by capturing the appropriate context. Also, the vanishing gradient problem in RNNs has been addressed successfully by LSTMs[4][10]. The context of a word depends on the words occurring before the word under consideration. To model this scenario, an LSTM based network is constructed. The LSTM design comprises a set of repetitively associated subnets, known as memory blocks.

Each block contains at least one self-associated memory cells along with three multiplicative units - the input, the output, and the forget gates - that give regular functionality of write, read and reset operations to the cells.

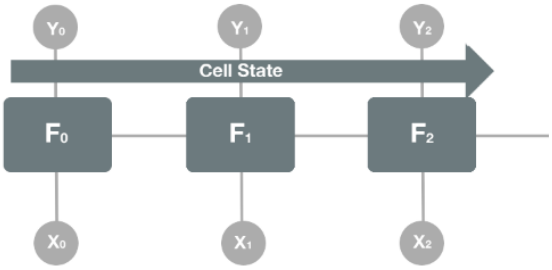


Fig. 3.3.2.1 LSTM architecture with cell state.

Figure 2 shows that LSTM has three components namely, forgot gate, an input gate, and an output gate. We made our LSTM model consisting of 5 layer architecture with 1 Embedding layer of shape (16,8), an LSTM layer of 64 nodes, two dense layers each made up of 64 and 1 output node, with a dropout layer, with 0.5 dropout parameter, in between them.

3.3.3 Bi-directional LSTM Model: Similar to LSTMs, Bi-directional LSTM models are also used to capture the context for text classification such as Text Generation, Sentiment Analysis or Sarcasm detection, etc. where the context of a word not only depends on words occurring after it. Modelling this requires memory cells in a backward direction which maintains the history of words along with cells in a forward direction for words not yet explored.

To achieve this and capture composition semantics of English and Hindi sentences, two bi-directional LSTM layers need to be used together with the first layer having an argument return_sequence = True.

We made a 6-layer model, the first layer for the embedding of size (16,8), the second one a bidirectional layer of 64 nodes, third also a bidirectional layer with 32 nodes, fourth and sixth dense layers with 64 and 1 node each in between connected via a dropout layer with dropout ratio of 0.5.

3.3.4 Multi-layered Bi-directional LSTM: We have stacked up two bi-directional LSTM layers to make it a multi bi-directional LSTM architecture. The first layer used was the embedding layer of size (16,8). The second layer was a bi-directional LSTM layer consisting of (64+64)=128 nodes with an argument, return_sequence equals True. The third layer is also a bidirectional layer of (32+32)= 64 nodes. The fourth and sixth are dense layers with 64 and 1 node each in between connected via a dropout layer, the fifth layer of our model, with a dropout ratio of 0.5.

4. RESULTS

4.1. Result Analysis: For English Sentences

A. RNN Model

Table 1: To analyze the overall accuracy and the validation accuracyover 10 epochs.

Model	Epochs	Accuracy	Validation Accuracy
RNN	1	53.02	56.55
	2	56.06	56.49
	3	56.31	56.48
	4	56.77	56.85
	5	58.24	59.99
	6	63.92	67.09
	7	68.66	69.99
	8	71.82	71.59
	9	73.70	73.14
	10	75..37	74.17

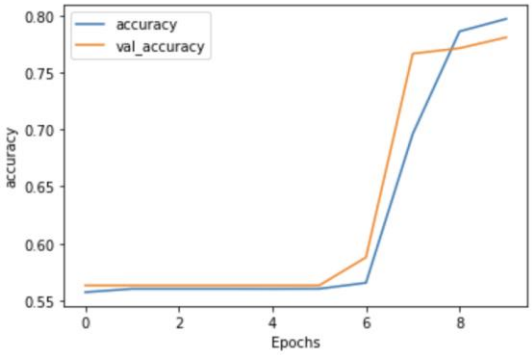


Fig. 4.1.A.1 Graph showing accuracy per epoch for RNN Model

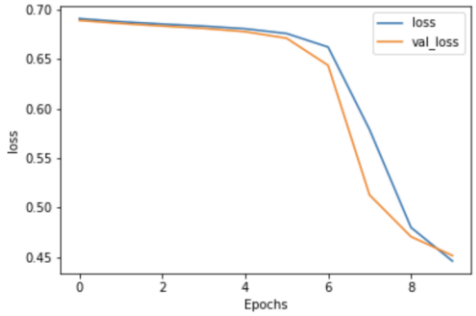


Fig. 4.1.A.2 Graph showing loss per epoch for RNN Model

B. Singled layered LSTM Units

Table 2: To analyze the overall accuracy and the validation accuracyover 10 epochs.

Model	Epochs	Accuracy	Validation Accuracy
LSTM units (64 units)	1	55.33	56.33
	2	56.04	56.33
	3	56.03	56.33
	4	56.03	56.33
	5	56.03	56.33
	6	56.03	56.33
	7	56.34	59.76
	8	69.21	74.66
	9	77.69	76.38
	10	79.22	77.51

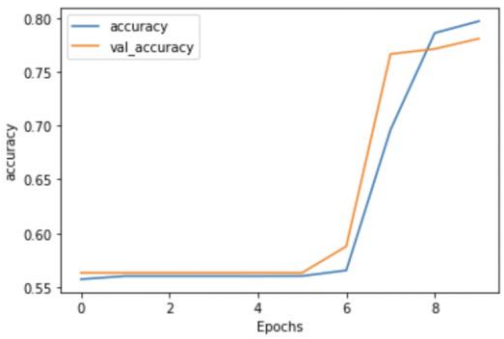


Fig. 4.1.B.1 Graph showing accuracy per epoch for LSTM Model

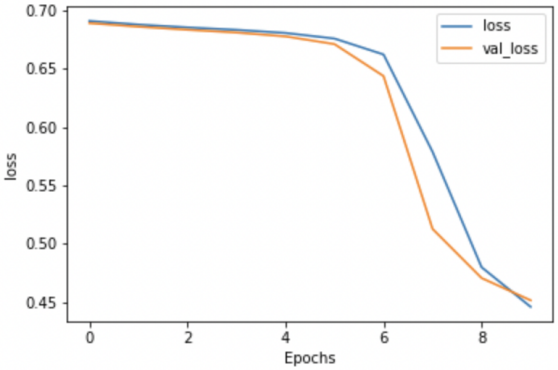


Fig. 4.1.B.2 Graph showing loss per epoch for LSTM Model

C. Singled Layered Bidirectional LSTM Units

Table 3: To analyse the overall accuracy and the validation accuracyover 10 epochs.

Model	Epochs	Accuracy	Validation Accuracy
Bi-Dir LSTM units (64 + 64 = 128 units)	1	55.73	56.33
	2	56.03	56.33
	3	56.03	56.33
	4	56.03	56.33
	5	56.02	56.33
	6	56.04	56.33
	7	56.56	58.79
	8	69.61	76.64
	9	78.60	77.12
	10	79.69	78.07

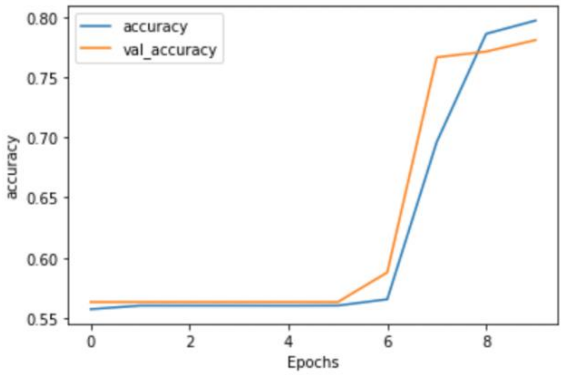


Fig. 4.1.C.1 Graph showing accuracy per epoch for Bi-directional LSTM Model

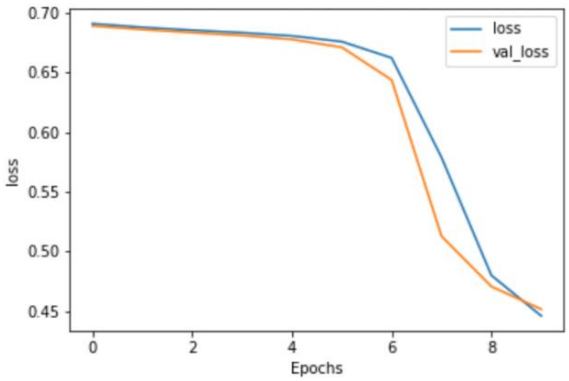


Fig. 4.1.C.2 Graph showing loss per epoch for Bi-directional LSTM Model

D. Multi Layered bidirectional LSTM Units

Table 4: To analyse the overall accuracy and the validation accuracyover 10 epochs.

Model	Epochs	Accuracy	Validation Accuracy
Multi-Layered Bi-Dir LSTM units (64 + 64 = 128 units and 32 + 32 = 64 units)	1	55.46	56.33
	2	56.03	56.33
	3	56.03	56.33
	4	56.04	56.37
	5	62.09	71.80
	6	75.56	75.18
	7	77.44	75.85
	8	78.85	77.15
	9	80.11	77.60
	10	81.16	78.48

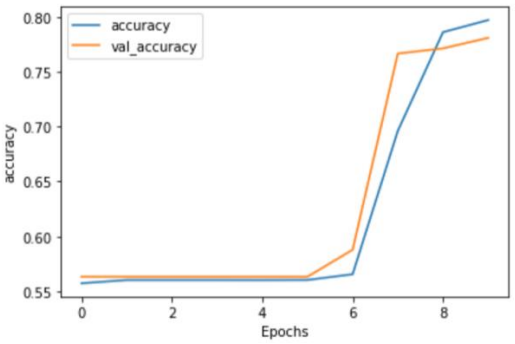


Fig. 4.1.D.1 Graph showing accuracy per epoch for Multi-layered Bi-directional LSTM Model

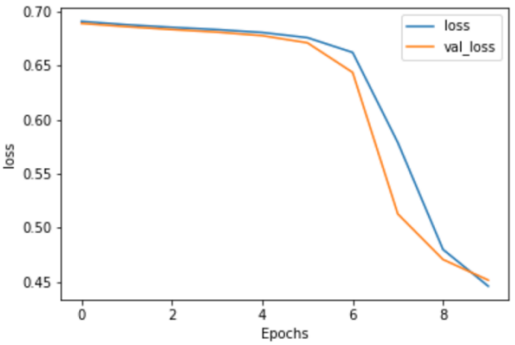


Fig. 4.1.D.2 Graph showing loss per epoch for Multi-layered Bi-directional LSTM Model

4.2. Result Analysis: For Hindi Sentences

A. RNN Model

Table 1: To analyze the overall accuracy and the validation accuracyover 10 epochs.

Model	Epochs	Accuracy	Validation Accuracy
RNN units (64 units)	1	68.21	00.00
	2	84.25	00.00
	3	84.40	00.00
	4	88.35	37.02
	5	91.91	57.36
	6	93.39	68.39
	7	93.92	74.37
	8	94.37	76.53
	9	94.83	78.20
	10	95.67	83.51

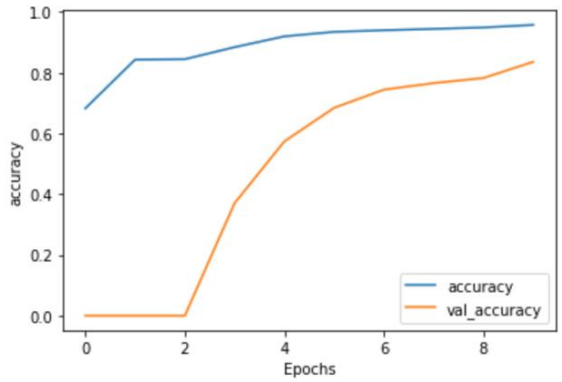


Fig. 4.2.A.1 Graph showing accuracy per epoch for RNN Model

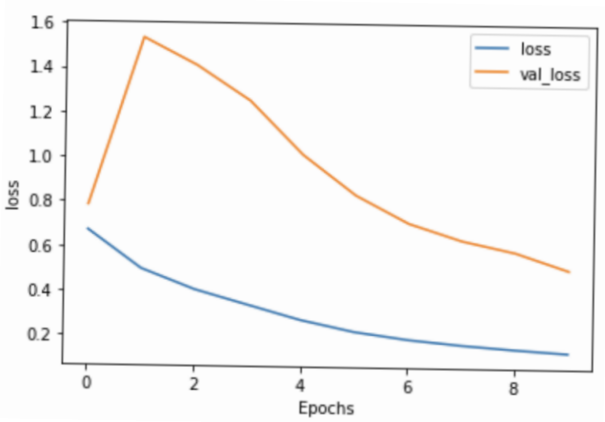


Fig. 4.2.A.2 Graph showing loss per epoch for RNN Model

B. Single LSTM Unit

Table 2: To analyze the overall accuracy and the validation accuracyover 10 epochs.

Model	Epochs	Accuracy	Validation Accuracy
LSTM Units (64units)	1	79.19	00.00
	2	84.40	00.00
	3	84.40	00.00
	4	84.40	00.00
	5	84.40	00.00
	6	84.40	00.00
	7	84.72	35.85
	8	90.70	52.05
	9	92.71	52.38
	10	93.41	61.43

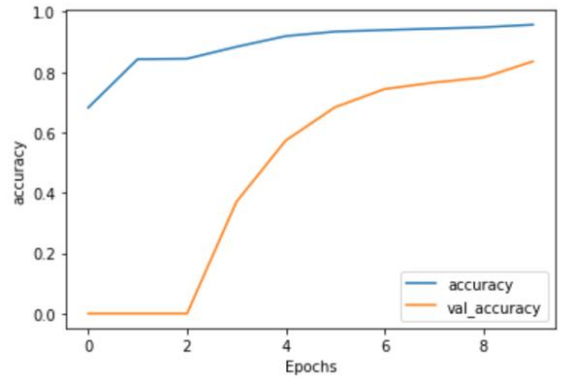


Fig. 4.2.B.1 Graph showing accuracy per epoch for LSTM Model

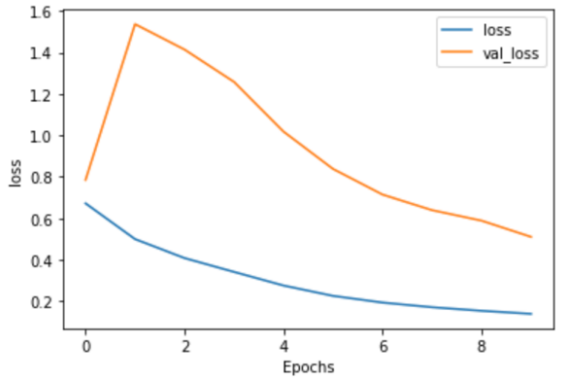


Fig. 4.2.B.2 Graph showing loss per epoch for LSTM Model

C. Singled Bidirectional LSTM Unit

Table 3: To analyze the overall accuracy and the validation accuracyover 10 epochs.

Model	Epochs	Accuracy	Validation Accuracy
Bi-Dir LSTM unit (64 + 64 = 128 units)	1	81.63	00.00
	2	84.40	00.00
	3	84.40	00.00
	4	84.40	00.00
	5	84.40	00.00
	6	84.40	00.00
	7	84.40	00.00
	8	89.04	39.87
	9	92.53	52.55
	10	93.20	61.59

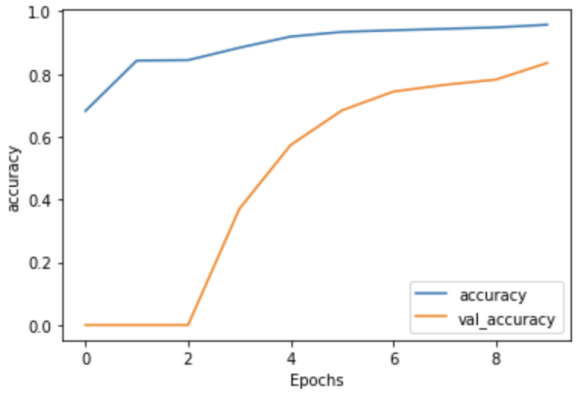


Fig. 4.2.C.1 Graph showing accuracy per epoch for Bi-directional LSTM Model

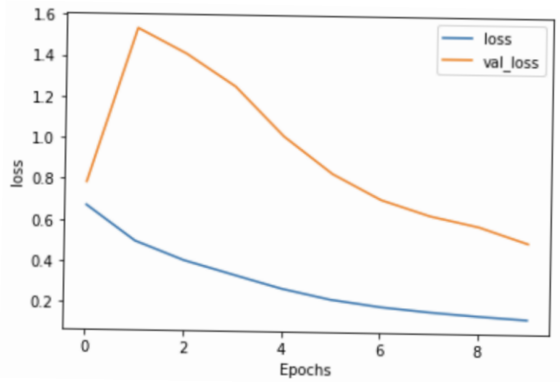


Fig. 4.2.C.2 Graph showing loss per epoch for Bi-directional LSTM Model

D. Multi Layered bidirectional LSTMUnits

Table 4: To analyze the overall accuracy and the validation accuracyover 10 epochs.

Model	Epochs	Accuracy	Validation Accuracy
Multi-Layered Bi-Dir LSTM units (64 + 64 = 128 units and 32 + 32 = 64 units)	1	83.47	00.00
	2	84.40	00.00
	3	84.40	00.00
	4	84.40	00.00
	5	84.40	00.00
	6	84.40	00.00
	7	85.07	35.85
	8	92.04	52.55
	9	93.41	59.42
	10	94.48	68.60

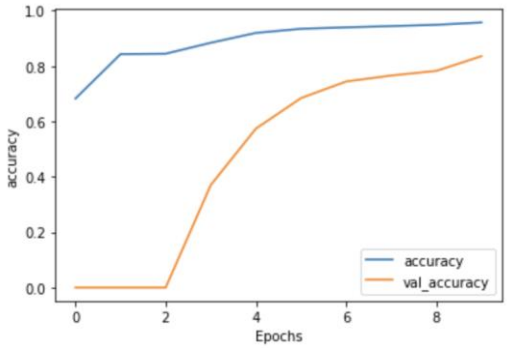


Fig. 4.2.D.1 Graph showing accuracy per epoch for Multi-Layered Bi-directional LSTM Model

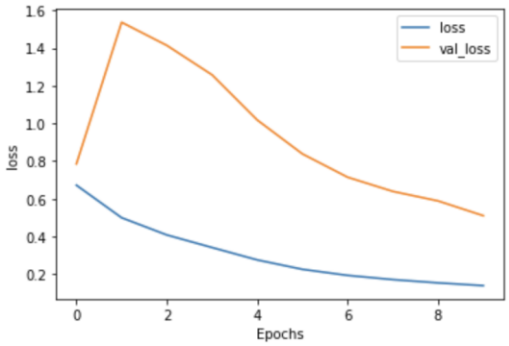


Fig. 4.2.D.2 Graph showing loss per epoch for Multi-Layered Bi-directional LSTM Model

5. CONCLUSION

In this paper, we compared the results of different LSTM and RNN models, when used to detect sarcasm in English and Hindi sentences separately. We saw how different LSTM and RNN models behave when we give the text in a language, i.e., Hindi, which they don't understand as fluently as they understand text written in the English language. We analyzed our input over four models named, Simple RNN[2], LSTM[4][5]][7], Bi-directional LSTM[9], and Multi-layer Bi-directional model. When we use them on English text, they gave us pretty decent results with an accuracy close to 80% and validation accuracy close to 78% in each of the four models we used. But when it comes to the Hindi text, we find very abnormal curves especially in validation accuracy when it's not showing any result till the 6th epoch. Thus, these models can efficiently be used to detect sarcasm in English text but, the models when used in their pure form over any foreign language, other than English, might not perform very well. Thus, there is a need of developing a combination of these models or others that could be able to detect sarcasm over any other language than English.

6. FUTURE SCOPE

The most important thing we can do is to get a well-versed dataset containing Hindi sentences, upon which we can make a Word2Vec or FastText model. And those word embeddings can be used as input for a more advanced model such as LSTM. That way we surely get better results, as by creating a word embedding beforehand, we training the model about the context that each Hindi word gives just like English words have distinct or similar context across the board.

7. REFERENCES

[1] Jenq-Haur Wang, Tinf-Wei Liu, Xiong Luo, Long Wang, "An LSTM Approach to Short Text Sentiment Classification with Word Embeddings", Conference on Computational linguistics and Speech Processing, 2018.

[2] Toma´s Mikolov, Martin Karafiat, Luka´s Burget, Jan "Honza" Cernock,Sanjeev Khudanpur, "Recurrent neural network based language model", ISCA,2010.

[3] Kaggle:[Online]
a.English: https://www.kaggle.com/rmisra/news-headlines-dataset-for-sarcasm-detection
b. Hindi: https://www.kaggle.com/pragyakatyayan/hindi-tweets-dataset-for-sarcasm-detection

[4] Sepp Hochreiter, Jurgen Schmidhuber, "Long Short-Term Memory", Neural Computation, 1997.

[5] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. Transactions of the Associa- tion for Computational Linguistics, 5:135–146.

[6] Adithya Rao and Nemanja Spasojevic. 2016. Action- able and political text classification using word embeddings and lstm. *ArXiv*, abs/1607.02501

[7] [Online] https://colah.github.io/posts/2015-08-Understanding-LSTMs

[8] T. Young, D. Hazarika, S. Poria, and E. Cambria. 2018. Recent trends in deep learning based natural lan- guage processing [review article]. IEEE Computa- tional Intelligence Magazine, 13(3):55–75.

[9] Basaldella, Marco & Antolli, Elisa & Serra, Giuseppe & Tasso, Carlo. (2018). Bidirectional LSTM Recurrent Neural Network for Keyphrase Extraction. 10.1007/978-3-319-73165-0_18.

[10] Hochreiter, Sepp. (1998). "The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions." International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems. 6. 107-116.10.1142/S0218488598000094.

