

# Advanced Combined Bidirectional LSTMs-DNN for Sarcasm Analysis on Twitter Dataset

Aishwarya S Muchandi  
Department of Computer Engineering  
A P Shah Institute of Technology  
Thane (Maharashtra), India  
aishum21102013@gmail.com

Prof. Jaya D Gupta  
Department of Computer Engineering  
A P Shah Institute of Technology  
Thane (Maharashtra), India  
jdgupta@apsit.edu.in

**Abstract**—In this paper, we are proposing an advanced combination of Bidirectional LSTMs (Long Short Term Memory) and DNNs (Dense Neural Network). In today's digital world, people are expressing their opinion and thoughts onto social media. And hence it is more important to understand and analyse the online data existing on various online webservices. The major challenge here is the size of such data. Hence, pre-processing of such raw data is a must! It has become extremely necessary for the organisations to understand the overall sentiments from these platforms and take necessary, accurate business decisions. In NLP (Natural Language Processing) analysis, it has further become an important aspect to interpret complex emotions such as Sarcasm which can be used for taking better business decisions. Sarcasm is an emotion which conveys a complete opposite message of what is actually being said. It's an ironic remark often used to express contempt or ridicule. The real meaning of sarcasm is hidden behind the meaning of sentences. Hence, the key feature of detecting sarcasm is by detecting the sudden change of sentiments in the message. Sarcasm is often expressed as a positive or negative sentiment followed by a twist of introducing the opposite sentiment in the message. Hence, to understand this change of sentiments, we are proposing Bidirectional LSTMs-DNN. In this model, Bidirectional LSTMs traverses and remembers the data sequence from both ends i.e. Forward and Backward information of the sequence. And Fully connected Layer i.e. Dense Layer can catch and learn information sufficiently. This combined model helps in understanding the sentiments from both sides of data sequence and computing the overall positive or negative emotion of data. This computing helps us in analysing if a sentence is sarcastic or non-sarcastic statement. If, overall positive emotion is computed, then the computing value is close to '0' and is termed as "Non-sarcastic" statement, as there is no negative sentiment detected. Similarly, if, overall negative emotion is computed, then the computing value is close to '1' and is termed as "Sarcastic" statement, as negative sentiments are detected. This type of analysis can be a ground-breaking breakthrough in NLP. This model can be used for many other fields in ML besides sarcasm detection. Such analysis can help the current digital industry working on online data to get better interpretation of data and as well as avoid misleading misunderstandings that might result into failure of business decisions.

**Keywords**—Sarcasm, Digital world, Natural Language Processing (NLP), data mining; sarcasm detection; Machine Learning (ML); Bidirectional Long Short Term Memory (LSTMs); Dense Neural Networks (DNN)

## I.

## INTRODUCTION

In today's world, huge online data is generated through various social sites such as Twitter, You-tube, LinkedIn, Facebook and many more. Today, Useful data holds ultimate power. Hence, such raw data obtained from various sites needs to be pre-processed in order to smoothly analyse the useful data. In a democratic world such as ours, many people like to express their thoughts through such online sites. People convey their emotions and perspectives through tweets, e-mails, messages. Many businesses run on such data such as Media Industry, Marketing Industry, Entertainment Industry, Foreign exchange industry. In addition to it, when a new product is launched in the market, the analysing committee sits to see various reactions and formulize next business strategy in order to increase company sales. Hence, such online data must be carefully refined and pre-processed before it is taken into consideration for accurate business decisions. For such online data analysing, Natural Language Processing is the best way of interpreting data from reactions that are expressed in human natural language. In NLP, pre-processing of raw data into useful data, avoiding stop words and interpreting the meaning of root word is possible.

People express their thoughts and reaction online. This reaction varies from person to person. This reaction conveys the emotions or sentiments that a person wants to convey. People have also invented lots of abbreviations for many words and emojis for expressing their sentiments. These emojis play a vital role in predicting sentiments. Sentiments can be happy, sad, angry, calm. These are all easy and simplified emotions to detect. But complex emotions such as Sarcasm is very difficult to predict, given the way they are expressed.

Sarcasm is a complex emotion where the person means completely opposite of what is being is actually being said. Sarcasm emotion can be easily recognized in verbal communication as even the voice tone is taken into consideration. But in online communications where only textual data is available devoid of sound analytics, it's very difficult to detect these emotions. Sarcasm is often used to express contempt and ridicule without the use of harsh negative words.

In fact, the real meaning of sarcasm is in between the meaning of statements. Sarcasm is often expressed by sudden change of sentiments. Sarcasm is often expressed as a positive or negative sentiment followed by a twist of introducing the opposite sentiment in the message. **For example:**

- 1) I happily spent \$500 to look this boring.
- 2) Are you always this annoying or are you exerting extra effort today?

Now, considering the first example: For a person using a company's product worth \$500, he means that the product is total waste. But the company's analytics lacking our model would detect it as a positive happy statement (because of "happily") giving in wrong information. Such analysis goes futile as analysis is made on incorrect information. Hence, it's very important to detect such emotions or sentiments in textual analysis.

The key feature of Sarcasm detection is detecting the sudden change of sentiments. Our advanced Model of Bidirectional LSTMs and CNNs exactly detects this sudden change of sentiments. The Bidirectional LSTMs traverses and analyses the data sequence from both the ends. It does Forward Traversing as well as Backward Traversing. While traversing, it recognizes the simple positive and negative sentiments and remembers it in the form of net value. Hence, even if initially the statement expresses "positive" sentiment and the ending expresses "negative" sentiment, our model can predict it as "Sarcastic" due to Bidirectional LSTMs which otherwise using normal neural networks is difficult.

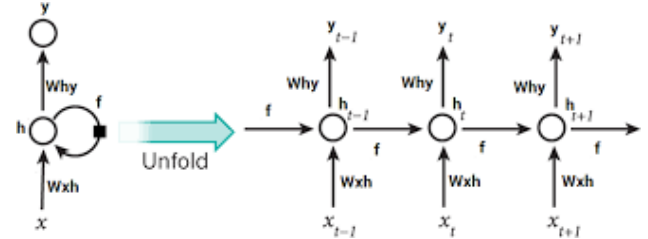
In this paper, we focus on detecting Sarcasm on Twitter Datasets. We are categorizing tweets into "Sarcastic" and "Non-Sarcastic". Our model recognizes "Sarcastic" as "0" and "Non-Sarcastic" as "1". This model can also be used for any other Textual Analysis in NLP for classifying purpose.

## II. EXISTING SYSTEM

### 2.1) Recurrent Neural Networks

A recurrent neural network (RNN) is a sub-class of artificial neural network wherein connections between units form a directed graph along a sequence. This lets it to showcase temporal dynamic behaviour for a time sequence. [7] Unlike feedforward neural networks, RNNs can use their internal state (memory) to manner sequences of data inputs. Hence, they are applicable to tasks such as unsegmented handwriting recognition or speech recognition. The term "recurrent neural network" is used indiscriminately to refer to two broad classes of neural networks 1) finite impulse and 2) infinite impulse, having similar general structure [6]. These both classes of neural networks exhibit temporal dynamic behaviour. A finite impulse recurrent neural network is a directed acyclic graph that allows unwinding and can be replaced with a strictly

feedforward neural network, while an infinite impulse recurrent neural network is a directed cyclic graph that cannot be unwound. Both finite impulse and infinite impulse recurrent neural networks may have additional stored state and its storage can be under direct control exhibited by the neural network. The storage can also be replaced by another neural network or graph, if it consists of time delays or feedback loops. Such controlled states are referred to as gated memory or gated state, and are part of Long short-term memories (LSTM) [1] and gated recurrent units.



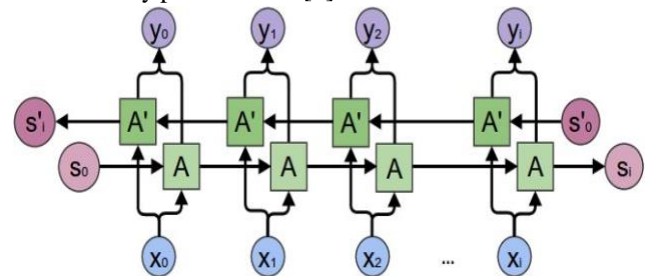
**Fig. 2.1.1 Recurrent Neural Network Logic**

RNN creates the networks consisting of loops, these loops allow it to carry the data information. This loop structure permits the neural network to take the data sequence as input. [7] RNN converts independent activations into dependent activations by providing the same weights and biases to all the layers; [as described above in Fig. 2.1.1]. Hence, reducing the complexity of increasing parameters by memorizing each previous output and giving it as input to the next hidden layer.

### 2.2) Bidirectional Long Short Term Memory (BiLSTM)

Bidirectional recurrent neural networks (RNN) are truly simply assembling two independent RNNs. This structure permits the neural networks to process both backward and forward information of the data sequence at every time step.[4],[8] Using bidirectional, will help traverse the input data sequence in two ways: 1] from past to future and 2] from future to past. Below Fig. 2.2.1 illustrates the working/traversing of BiLSTMs.

And what varies this methodology from unidirectional is that the LSTM predicts future values based on past information but cannot predict vice-versa. By using the two combined hidden states of BiLSTM; one can preserve information from both past and future at any point in time.[8]



**Fig. 2.2.1 Bidirectional Long Short Term Memory (BiLSTM) Logic**

BiLSTMs show very good results as they can understand the context better. Below is an illustrated example for reference: Here, we are trying to predict the next word in a sentence using unidirectional LSTM. The unidirectional LSTM will see:

Example:

“The girls went to ....”

It will try to predict the next word only by this context based upon the words traversed in the above example.

Whereas with bidirectional LSTM, we are able to process the information further down the road.

Example:

Forward LSTM:

“The girls went to ....”

Backward LSTM:

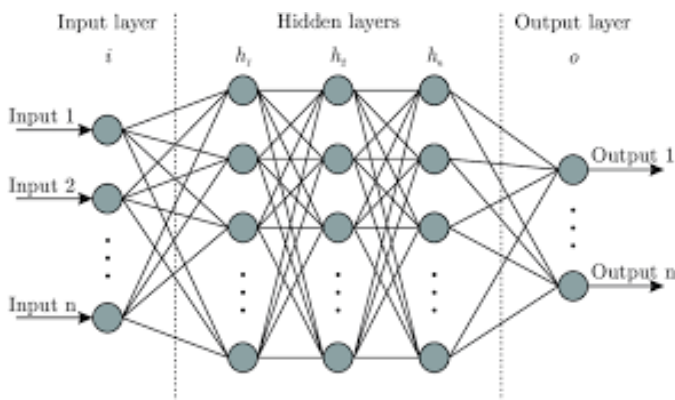
“... and then they got out of the pool.”

The information from the future is also being traversed and hence making it easier for the neural network to understand and predict the next precise word that would be a best-fit to the input data sequence. [4],[8]

Hence, we are using BiLSTMs to recognize the sentiments from both the ends of data sequence.

### 2.3) Dense Neural Network (DNN)

A densely connected layer provides learning features from all the combinations of the features of the previous layer, whereas a convolutional layer relies on consistent features with a small repetitive field. [6] This feature of learning features from all possible combination of features from previous state helps us in considering all the possible possibilities in detecting “sarcastic” or “non sarcastic” emotion. In text classification, one cannot determine which text can play important role, [6] as it really depends on the meaning that is being transferred through words. Hence, it’s impossible to determine the consistent important features of previous state. Dense Neural Networks are often called as Fully connected Layer.



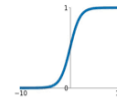
**Fig. 2.3.1 Dense Neural Network Graph**

Each neuron in a layer receives an input from all the neurons present in the previous layer—thus, they’re densely connected as illustrated in previous Fig. 2.3.1. Also, various **activation functions** and **weights** are used for each node.[6] These activation functions play an important role in learning of neural networks. Each activation function has a mathematical formula to compute weights [refer below Fig. 2.3.2], which are later assigned to various nodes of next hidden layer.

### Activation Functions

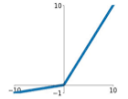
**Sigmoid**

$$\sigma(x) = \frac{1}{1+e^{-x}}$$



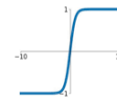
**Leaky ReLU**

$$\max(0.1x, x)$$



**tanh**

$$\tanh(x)$$

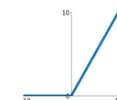


**Maxout**

$$\max(w_1^T x + b_1, w_2^T x + b_2)$$

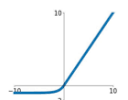
**ReLU**

$$\max(0, x)$$



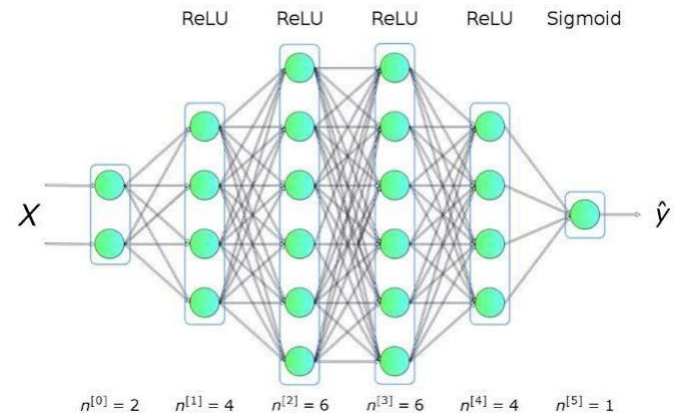
**ELU**

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



**Fig. 2.3.2 Different Activation Functions with their respective learning rate graphs and mathematical formulae.**

Also, every activation function has its own **learning rate**. This learning rate decides how quickly to learn important features from the data. A particular activation functions works best for a particular dataset solving particular problem. Hence, after many permutations and combinations of hyperparameter tuning, we have recognized the best combination of activation function that works fine with our model as described below in Fig. 2.3.3



**Fig. 2.3.3 Graphic Depiction of Activation Functions used in our model.**

### III.

### EXPERIMENT

There are in total of 36,423 tweets in the dataset classifying these tweets into “Sarcastic” and “Non-Sarcastic” statements. “Sarcastic” is denoted as “1” and “Non-Sarcastic” as “0”. In addition to it, we have considered 2 more datasets of emojis and abbreviations. In emojis dataset, the image of emoji as well as its meaning is mentioned and in abbreviations dataset,

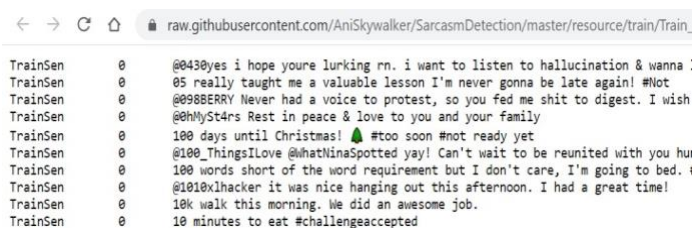
many abbreviations used in social world as shortcuts are expanded. These datasets consider every aspect of unprofessional communication such as abbreviations, emojis etc. Hence, all these datasets were taken into consideration while analysing these tweets.

### 3.1) Datasets

The links for dataset are as follows:

- 1) [https://raw.githubusercontent.com/AniSkywalker/SarcasmDetection/master/resource/train/Train\\_v1.txt](https://raw.githubusercontent.com/AniSkywalker/SarcasmDetection/master/resource/train/Train_v1.txt)
- 2) [https://github.com/AniSkywalker/SarcasmDetection/blob/master/resource/emoji\\_unicode\\_names\\_final.txt](https://github.com/AniSkywalker/SarcasmDetection/blob/master/resource/emoji_unicode_names_final.txt)
- 3) <https://github.com/AniSkywalker/SarcasmDetection/blob/master/resource/abbreviations.txt>

#### Screenshots of Datasets

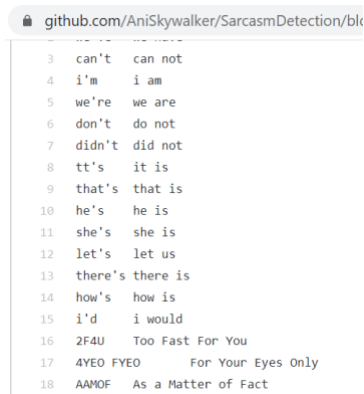


```

TrainSen 0 @0430yes i hope youre lurking rn. i want to listen to hallucination & wanna
TrainSen 0 05 really taught me a valuable lesson I'm never gonna be late again! #Not
TrainSen 0 @0988ERRY Never had a voice to protest, so you fed me shit to digest. I wish
TrainSen 0 @0HMySt4rs Rest in peace & love to you and your family
TrainSen 0 100 days until Christmas! 🎄 #too soon #not ready yet
TrainSen 0 @100_ThingsILove @ihatinasSpotted yay! Can't wait to be reunited with you hu
TrainSen 0 100 words short of the word requirement but I don't care, I'm going to bed.
TrainSen 0 @1010xlhacker it was nice hanging out this afternoon. I had a great time!
TrainSen 0 10k walk this morning. We did an awesome job.
TrainSen 0 10 minutes to eat #challengeaccepted

```

**Fig. 3.1.1** Tweet Dataset labelled as “0” (Non-Sarcastic) & “1” (Sarcastic)

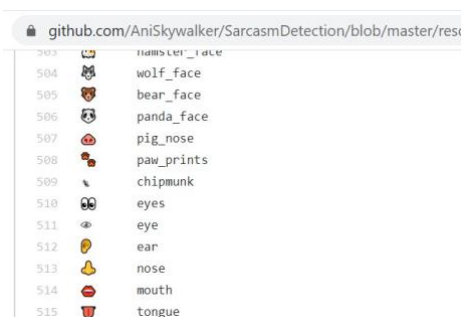


```

3 can't can not
4 i'm i am
5 we're we are
6 don't do not
7 didn't did not
8 tt's it is
9 that's that is
10 he's he is
11 she's she is
12 let's let us
13 there's there is
14 how's how is
15 i'd i would
16 2F4U Too Fast For You
17 4YEO FYEO For Your Eyes Only
18 AAMOF As a Matter of Fact

```

**Fig. 3.1.2** Abbreviations dataset



```

503 🐼 hamster_face
504 🐺 wolf_face
505 🐻 bear_face
506 🐼 panda_face
507 🐷 pig_nose
508 🐾 paw_prints
509 🐿 chipmunk
510 👁 eyes
511 👁 eye
512 👂 ear
513 👃 nose
514 👄 mouth
515 🐸 tongue

```

**Fig. 3.1.3** Emojis Dataset

The first most initial step is preprocessing of this data, which was carried out in 4 steps:

1. **Tokenization**: Tokenizing sentences into words.[2]
2. **Removal of Stopwords**, punctuations and usernames.
3. **Stemming and Lemmatization**: Conversion of words into their root words, hence reducing the dataset to a great extent. [3]
4. **POS Tagging**: It helps us to classify the words into a verb, adverb, adjective, noun etc. So that when there is too much of sarcasm then that leads to a hint of the presence of sarcasm i.e., by providing too much of praise in the sentence.[2],[3]

**3.2) Feature Extraction**: There are various ways of extracting feature and then determine the important data. [3] The various ways are:

- 1) **Term Frequency**: Term-frequency determines the number of times a particular word appears in the document.
- 2) **Term Frequency-Inverse Document Frequency (TF-IDF)**: It is the frequency measure of occurrence of a feature in the document. The required features are extracted and vectorized using TF-IDF.[3]
- 3) **Weighted-Term-Frequency-Inverse-Document Frequency**: It is used in the calculation of sentiment score using slang words with the help of weighted TF-IDF is done.
- 4) **Delta Term Frequency-Inverse Document Frequency (TF-IDF)**: It makes an easy computation, understanding and implementation. SVM's are used to show delta IF-IDF increases the accuracy of sentiment analysis. It works on both sentiment polarity [2]
- 5) **Word2vec**: Word2vec produces a vector space and defines the representation of words.
- 6) **Pattern-Related**: The Supervised learning method which leans towards sarcastic patterns based on parts-of-speech. It the pattern of high frequency words. Length of the pattern is also taken into consideration. Generalized patterns are based on three situations namely exact match, no match and partial overlap.

We have used a combination of **Term Frequency**, **TF-IDF**, **Pattern Related**. This is done by using **CountVectorizer** function in scikit-learn. Overall, it is **Bag Of Words Model** that is used here. Here the document is an “input” and a class label is the “output”. Here any text can be encoded as a fixed-length vector with the length of the vocabulary of known words. [5] The value in each position in the vector could be filled with a count or frequency of each word in the encoded document. [9] Through this method, we formed a vocabulary of 1051 words which occurred more than 50 times.

Later, we used **Glove Embeddings** [5],[9] to map all our 1051 words with the words in Glove embedding file and formed a crisp data.



Word embeddings are a type of word representation that allows words with similar meaning to have a similar representation. Hence reducing the raw data to crisp data containing all the vectors of important root words as all the words having same meaning are mapped to single word and hence single vector representation.

**For example:**

- 1) “Man” and “King” will have similar vectors.
- 2) “Girl” and “Queen” shall have similar vector representation.
- 3) King – Man + Woman = Queen

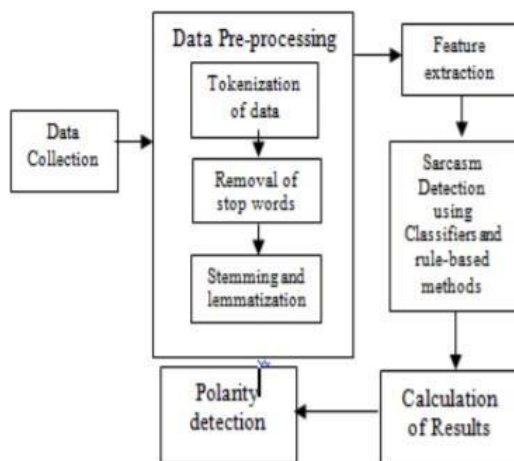
Word embedding methods learn a real-valued vector representation for a predefined fixed sized vocabulary from a corpus of text. [5] Global Vectors for Word Representation, or GloVe, algorithm is an extension to the word2vec method for efficiently learning word vectors.[9] GloVe constructs an explicit word-context or word co-occurrence matrix using statistics across the whole text corpus. The result is a learning model that may result in generally better word embeddings. Here the embedding dimension considered is 300. Later the data was splitted into 3 parts for further evaluation purposes. Refer below Table 3.2.1

**Splitting Data as follows:**

Splitting	Percent	Actual Values
Training	80%	29138
Validation	10%	3641
Testing	10%	3642

**Table 3.2.1** Splitted Data

Later we used our advanced model of BiLSTMs-DNN to train and predict Sarcasm of the given tweet. This model can be applied to all textual datasets to detect emotion (Sarcasm) analysis. Refer below Fig. 3.2.2 for understanding its workflow.



**Fig. 3.2.2** Detailed Workflow of sarcasm detection in sentiment analysis.

### 3.3) RESULTS

The model was trained using our advanced networks of BiLSTM-DNN. In addition to it we compared our model to LSTM-CNN. Also, the embedding layer is used in the front-end of a neural network and is fit in a supervised way using the Backpropagation algorithm; [as seen in below Fig. 3.3.1]. It is given an input text which is already cleaned and prepared, such that each word is one-hot encoded. The size of the vector space is specified as part of the model of 300 dimensions. Below Fig. 3.3.1 shows detailed statistics of calculated learnable parameters for each individual type of layer used in the model.

Layer (type)	Output Shape	Param #
input_10 (InputLayer)	(None, 50)	0
embedding_10 (Embedding)	(None, 50, 300)	315300
bidirectional_10 (Bidirectio	(None, 64)	85248
dense_29 (Dense)	(None, 16)	1040
dropout_1 (Dropout)	(None, 16)	0
batch_normalization_15 (Batc	(None, 16)	64
dense_30 (Dense)	(None, 8)	136
dropout_2 (Dropout)	(None, 8)	0
batch_normalization_16 (Batc	(None, 8)	32
dense_31 (Dense)	(None, 4)	36
dropout_3 (Dropout)	(None, 4)	0
dense_32 (Dense)	(None, 1)	5
Total params:		401,861
Trainable params:		86,513
Non-trainable params:		315,348

**Fig. 3.3.1** Model Fitting

After a lot of permutations and combinations of Hyperparameter Tuning using GridSearchCV, we reached the final values that perfectly fits to our neural networks, giving accuracy of 90%. Below Table 3.3.2 shows final values set after successful conduction of Hyperparameter Tuning

Parameters	Values
Embedding Dimension	300
Epochs	10
Activation	Relu, Sigmoid
Optimizer	Adam
Dropout	0.4

**Table 3.3.2** Final Parameters after Hyper-Parameter Tuning.

## Results Analysis

Our model has reached 90% accuracy level with the advanced combined model of BiLSTMs and DNN. We compared our model to LSTM-CNN. Our model outperforms the other model by 13.7%. Hence, our model indicates that it improves the accuracy level due to accurate feature extraction and effective learning. These results show that our initial intuition of using BiLSTMs can promote to effective and accurate decision computation of detecting sarcasm as “1” and non-sarcasm as “0”.

Grouping” 2018 International Joint Symposium on Artificial Intelligence and Natural Language Processing (iSAI-NLP).

## IV.

## CONCLUSION

Our model of BiLSTMs-DNN serves as the best method to interpret complex emotions such as “Sarcasm”. Sarcasm as we had defined earlier is expressed as a positive or negative sentiment followed by a twist of introducing the opposite sentiment in the message. The key feature is to detect this sudden change of sentiments in a sentence. Our Bidirectional LSTMs traverses the sequence of data from both the ends i.e. Forward Traversing as well as Backward Traversing and hence can remember forward information as well as backward information. While traversing, it recognizes the simple positive and negative sentiments and computes the sentiment polarity. This is the key advantage BiLSTM have over LSTM which leads to accurate sarcasm detection. Besides our Fully Connected layer i.e. Dense Layer catches and learns all the information efficiently. This advanced model will help our industries grow, improve at their products and provide better customer satisfaction. And our model is not restricted to only sentiment analysis, it’s a good model which can be used in classification model in other machine learning fields. In the future, we can link it to other NLP technology (yet to be discovered) to give better results.[1]

## V.

## REFERENCES

1. Nan Chen, Peikang Wang: “Advanced Combined LSTM- CNN model for twitter sentiment analysis” 2018 5th IEEE International Conference on Cloud Computing and Intelligence Systems (CCIS).
2. Sneha Pasarate, Rajashree Shedge: “Comparative study of feature extraction techniques used in sentiment analysis” 2016 International Conference on Innovation and Challenges in Cyber Security (ICICCS-INBUSH)
3. Kwanrutai Nokkaew, Rachada Kongkachandra: “Keyphrase Extraction as Topic Identification Using Term Frequency and Synonymous Term

4. Jun Xie , Bo Chen , Xinglong Gu , Fengmei Liang, Xinying Xu : “Self-Attention-Based BiLSTM Model for Short Text Fine-Grained Sentiment Classification” IEEE Access (Volume 7).
5. Rahul S. Dudhabaware , Mangala S. Mandekar: “Review on natural language processing tasks for text documents” in 2014 IEEE International Conference on Computational Intelligence and Computing Research.
6. Dionysis Goularas , Sani Kamis: “Evaluation of DeepLearning Techniques in Sentiment Analysis from Twitter Data” in 2019 International Conference on Deep Learning and Machine Learning in Emerging Applications (Deep-ML).
7. Akanksha Rai Sharma, Pranav Kaushik: “Literature survey of statistical, deep and reinforcement learning in natural language processing” in 2017 International Conference on Computing, Communication and Automation (ICCCA).
8. Abdullah Aziz Sharfuddin ; Md. Nafis Tihami ; Md. Saiful Islam “A Deep Recurrent Neural Network with BiLSTM model for Sentiment Classification” in 2018 International Conference on Bangla Speech and Language Processing (ICBSLP).
9. Wei Yen Chong ; Bhawani Selwaretnam ; Lay-Ki Soon: “Natural Language Processing for Sentiment Analysis: An Exploratory Analysis on Tweets” in 2014 4th International Conference on Artificial Intelligence with Applications in Engineering and Technology.

- [1] Che D., Safran M., Peng Z. (2013) From Big Data to Big Data Mining: Challenges, Issues, and Opportunities. In: Hong B., Meng X., Chen L., Winiwarter W., Song W. (eds) Database Systems for Advanced Applications. DASFAA 2013. Lecture Notes in Computer Science, vol 7827. Springer, Berlin, Heidelberg. [https://doi.org/10.1007/978-3-642-40270-8\\_1](https://doi.org/10.1007/978-3-642-40270-8_1)

