International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-10 Issue-1, October 2020

Sarcasm Detection of Sentiments in Telugu Language

Suneetha Eluri, Naga Santosha Lahari Penmatsa

Abstract: ***Sarcasm is usually used by people to either tease/irritate others or simply for comic purposes. The presence of sarcasm becomes certain as it is difficult to be identified by basic sentiment analysis method. Sarcasm detection is addressed with various rule-based methods, statistical approaches, and classifiers in machine learning , most of these are introduced to identify sarcasm in text written in English as it is a popular language on the internet. Although the groundwork done on sarcasm detection on various Indian languages like Telugu is limited. Hence, this paper presents a Deep learning model based on neural networks to detect sarcasm in Telugu news headlines taken from various websites . The proposed model comprises of Convolutional Neural Networks(CNN) and next a Long short-term memory(LSTM) Network which is a modified version of Recurrent neural networks (RNN) and lastly a fully connected dense layer is added to classify the sentiments into sarcastic and non-sarcastic. A pre-trained word embeddings GloVe are used in the model***

Keywords : ***Convolutional Neural Networks, Deep learning, Long-short term memory, Sarcasm.***

I. INTRODUCTION

Sarcasm Detection is one of the research field in Natural language processing (NLP), a distinct case which is part of sentiment analysis where rather than identifying the sentence in the text as positive or negative sentiment, the emphasis is on sarcasm. Sentiment analysis is an approach that examines individual's opinions, thoughts ,feelings towards a target[1]. The existence of sarcasm in the text make sentiment analysis task strenuous as sarcasm changes the polarity of the text.

Sarcasm is when people say something inconsistent from what they mean. It is also stated that sarcasm happens when positive sentiment coupled to a negative situation[2]. Consider a simple example: " I love waiting for the bus for long hours # sarcasm". It is seen that in the above example the word 'love' implies a positive sentiment, but the situation in the example is negative as "nobody likes to wait for long hours". This implies that the above example is sarcastic. By the usage of grammar, lexical structures and contextual information in text, sarcasm detection is performed[3]. Identifying sarcasm is easy while people communicating orally but it is challenging task to identify it in text.

Revised Manuscript Received on October 25, 2020.
* Correspondence Author
Dr. Suneetha Eluri*, Professor, CSE Department, JNTUK, Andhra Pradesh, India. Email: suneethaeluri83@gmail.com
Naga Santosha Lahari Penmatsa, Academics, CSE Department, JNTUK, Andhra Pradesh, India. Email: lahari.penmatsa@gmail.com

As internet is growing day by day and people expressing their opinions through various social networking sites (SNS) and websites in various languages, this led us to work on this domain(sarcasm detection for Telugu language) as Telugu is one of the popular language in India, as there is famous saying by south Indian king Srikrishna Devaraya that "of all the languages in this country Telugu language is the best". An automatic sarcasm detector when present in sentiment analysis system will improve decision making process in many aspects.

This paper concentrates on detecting sarcasm on corpus containing News headlines written in Telugu language . The main focus lies on understanding the challenges, issues while working on Telugu language and the approaches followed while performing classification (sarcastic or non-sarcastic) for a particular sentence in text taken in Telugu language. With very few tools and annotated corpora available, it is a challenging task to perform sarcasm detection for Telugu language.

II. LITERATURE SURVEY

This sectional survey gives an overview of previous work done on sarcasm detection in different languages like English and other local languages.

Diana Maynard and Mark A Greenwood [4] exhibited the significance of sarcasm detection in sentiment analysis . In this experimentation they stated that there is a huge development on the result by utilizing sarcasm detector in sentiment analyzer system. Hashtags were used to detect sarcasm, but just depending on hashtags is inefficient as there can be cases where no hashtags are used in sentence containing sarcasm.

P. Dharwal et al. [5] considered the techniques that are deployed for automatic sarcasm detection. They have observed that n-grams when solely used are not enough for accurate classification but combining these with further methods can help in the rise of accuracy. It is seen that SVM method is more efficient than Logistic regression and Naive bayes.

A pattern-based method was used to detect sarcasm in twitter and classify the tweets into sarcastic and non-sarcastic. M. Bouazizi and T. O. Ohtsuki [3] evaluated the usage of sarcasm in 3 ways: the first usage is when a person tries to act funny, another usage is when he/she is in anger, the last usage is when he/she wants to ignore answering. The feature set is derived based on the above assumptions. They have used classifiers like "SVM", "Random forest", "maximum entropy",

"K-nearest neighbor" for classification. Accuracy of 83.1% is obtained from this method.

Rajadesingan et al. [6] did research on the twitter user writing style for sarcasm detection. They mentioned the complications in the nature of the twitter data for sarcasm detection and showed the importance of twitter user's history for sarcasm detection but this method is less efficient when there is no previous information of the user.

Poria et al. [7] proposed models which depend on pre-trained convolutional neural network for extracting features for sarcasm detection . For classification both CNN and CNN-SVM are used. In CNN-SVM, the classification results of CNN are given to SVM for final result. It is seen that CNN-SVM performs better than CNN in all aspects.

Le Honag Son et al. [8] applied both Soft attention-based bidirectional long short-term memory(sAtt-BLSTM) and convolutional neural networks on SEMEVAL 2015 TASK 11 dataset and random tweets dataset. GloVe word embeddings are given as input to this deep learning model. The feature maps obtained from sAtt-BLSTM are given as input to CNN network for obtaining final results. An accuracy of 97.87% is achieved by this model with SEMEVAL 2015 task 11 dataset and 93.71% obtained from random tweets dataset.

Yessi Yunitasari et al. [9] applied machine learning algorithms like random Forest , Naive bayes for classification of Indonesian tweets. They have used unigram , TF-IDF approaches for feature extraction and also used Boazizi[3] four feature sets. This model which is used for sarcasm detection of sentiments obtained an accuracy of 80.4%, with precision and recall of 83.2% , 91.3% respectively.

Shih-kai lin and Shu-kai Hsieh [10] suggested a corpus generation procedure for Chinese Language by which we can collect social data. They have used machine learning technique SVM and Deep-learning model CNN for training of the data. It is observed that CNN model performs better than SVM, as CNN obtained an accuracy of 87% and SVM with 55% of accuracy.

D. Ghadhban et al. [11] trained the Arabic tweets dataset using a supervised naive bayes multinominal text algorithm and features are given to Weka classifier to detect sarcasm in the tweets. They have collected the tweets using hashtags. This model attained precision of 0.659, recall of 0.710 and f-score of 0.676.

Santosh Kumar Bharti et al. [12] initiated a context-based pattern which is used in Hindi tweets for sarcasm detection . They have considered the Hindi social media news from twitter sources as the context of the tweet in the similar timestamp. This method attained an accuracy of 87%.

III.PROPOSED SYSTEM

We have collected the text from news websites like eenadu.net, andhrajyothi.com, blog posts etc., written in Telugu language and saved in a csv file to perform sarcasm detection using deep learning model and to classify the text as sarcastic or non-sarcastic. The deep learning model shown in Fig.1 consists of following layers embedding layer (GloVe Embedding vectors are used),Conv1D layer, Maxpooling1D layer, LSTM layer, Dense layer and each layer has its own functionality for building the model. The above layers are provided by keras library in python. The Proposed Deep learning model is trained on the 70% of the data and then tested on the remaining 30% of validation set to calculate the accuracy of analyzing sentiment. This is done to train the model for accurate prediction. The input text undergoes preprocessing and converts into a vector that is understandable by the machine.

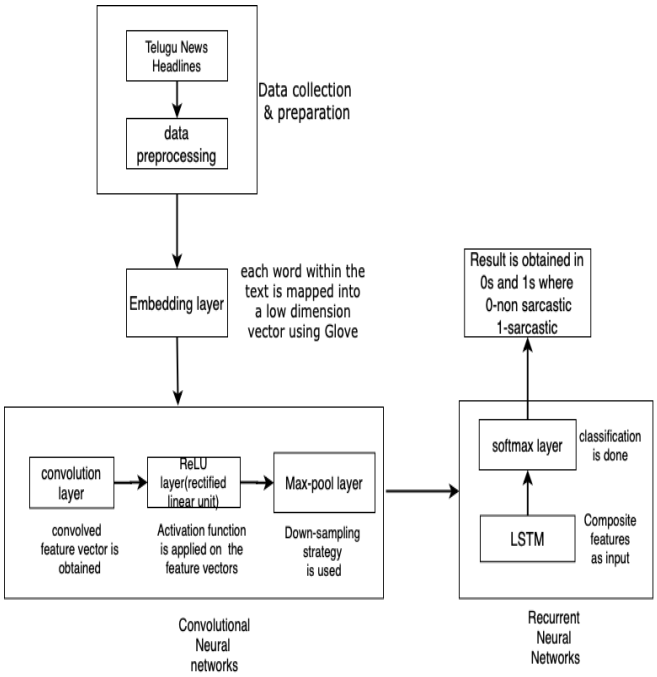

Fig. 1. Architecture of the Proposed Model

A. GloVe Embedding

GloVe model[13] let us to convert each word in the corpus of text into real-valued vectors known as embedding vectors. It is an open source project developed by Stanford. This is an unsupervised learning algorithm which helps us to place similar words together.

B. Convolutional Neural Networks

Convolutional neural networks[14] contain layers which help in extraction of the features in text. The Network is trained on the dataset to make the layers learn the features and finally a sentence to be classified when given as input, probabilities are obtained by the network. These probabilities help us to known which class a sentence in the text belongs to. After calculating the output, the error is computed. The error obtained is back propagated through the network to modify the parameters accordingly, so that the error can be reduced. This complete process repeats till the model operates well.

Following are the layers of Convolutional Neural Networks:

• Convolution Layer : This layer is considered as a significant in CNN as majority of operations are performed here . By building word embedding matrix from the text . The filter/Kernel(interchangeably we can use these) slides over the embeddings to form convolutions. This is essential in extraction of features. After applying the filter over the entire embedding matrix several feature maps are obtained.

For example, consider 64 filters of kernel size 2 on the layer that means 64 different features are trained [15].

• Activation Layer : This layer is used to produce non-linearity in the output. This non-linearity helps the neural network to make training more accurate and faster. Some of the activation functions which can be applied to

• CNN are tanh, Identity , Rectified Linear unit (ReLu).

For example, ReLu activation function converts all negative values to Zero

• Max-Pooling Layer : This layer is mainly used to reduce dimensionality of the feature maps obtained from convolutional layer still keeping the important information. Computational complexity is also reduced by pooling. For example, consider pooling size of 3 ,which means the size of output matrix is just a third of input matrix [15].

C. Long Short-term Memory (LSTM)

The LSTM model [16] is a modified version of RNN, made up of memory cells to keep the information for longer periods and gating units to decide what information to store and when to use that information. This model is specifically designed to remove long-term dependencies problem. Hence, despite of its complicated computation , the network provides good results since it can take whole sentence as context to obtain results in various NLP tasks [17][18]. LSTM are mainly designed to remove Vanishing Gradient problem which is a major problem in traditional RNN.

III. METHODOLOGY

This section introduces the methodology of the proposed system which is used for classifying the text whether it is sarcastic or non-sarcastic.

A. Dataset

• The dataset that was taken consists of Telugu news headlines, which are collected from Telugu news websites and it is uploaded to the google drive in order to work with google colab. It consists of columns like Id, heading, is_sarcastic.

• The heading column is the important column in the dataset in which it contains 4000 plus sentences out of which 3405 are sarcastic and 942 are non-sarcastic. Based on these sentences we have done this sarcasm classification.

• Each record consists of two attributes:
  is_sarcastic: 1 if the entry is sarcastic otherwise 0
  Heading : healine of the news article

• Python provides pandas Library which helps to use the data in the dataset.

B. Preprocessing

Online texts usually contain lots of noise and uninformative parts such as duplicate data(stop words), digits/numbers etc. In order to reduce the complexity for classification this unnecessary data should be removed from the data set. This process is known as text preprocessing.

Table-I: sample headline and processed headline

SAMPLE HEADLINE	ఇరాక్‌లో ఆత్మాహుతి దాడి 9 మంది మృతి (English version: Suicide bomber kills nine in Iraq)
PROCESSED HEADLINE	ఇరాక్ ఆత్మాహుతి దాడి మంది మృతి

C. Training of CNN-LSTM model

Train_test_split function from Scikit-learn slipt the data into training data and test data

1) Input layer

The sentences after pre-processing are given to the input layer, then sent to the Glove embedding model, for attaining vector representation for each word in the corpus.

2) GloVe embedding

A neural network cannot work directly on text-strings so we must convert it somehow. There are two steps in this conversion, the first step is called the “tokenizer” which converts words to integers and is done on the dataset before it is given as input to the neural network and padding is also performed to ensure that all sequences in the entire dataset have the same length as Neural Networks cannot work with the sequences of different lengths when it deals with a whole batch of data. The second step is to use a pre-trained word embeddings. The tokenizer can then be “fitted” to the dataset. Here we have instructed the tokenizer to only use the 10400 most popular words from the dataset. Took all the text in the dataset and called the function fit_on_texts on the tokenizer to build the vocabulary. Each word in the data set is associated with an integer known as tokens.

glove-python library in python is used in implementation of GloVe model**.** An array of words is taken as input to GloVe model. These words are stored in array hl_tokens.

Following are the steps for building a glove model:

Step 1: To build a glove model a corpus object is created.

corpus= Corpus()

Step 2: Next we train the corpus to create the co-occurrence matrix

corpus.fit(hl_tokens, window=10)

here hl_tokens is 2D array of words which is created after preprocessing task and window represents the space between two words.

Step 3: A glove object is created which makes use the above created matrix to generate embeddings.

glove= Glove(no_components=100,learning_rate=0.05)

here no_components represent the dimension of output vector which are created by the GloVe.

Step 4: Finally, we train the glove object by using the word to word co-occurrence matrix which is obtained by training the corpus object.

glove.fit(corpus.matrix,epochs=30,no_threads=4,verbose=True)

here epochs define how many times the algorithm should go through the dataset .

step 5: To get access to embeddings the dictionary should added to the glove object as the dictionary still exists in corpus object.

glove.add_dictionary(corpus.dictionary)

After completion of above steps , the glove object is set to provide you the word embeddings. Now an input dimension of 10400 is taken as we have already mentioned vocabulary size where integers are encoded from 0 to 10399 and defined output dimension to be 100.

finally, we will not again train the word weights learned from glove model in the neural network model, so the trainable element is set to False.

3) Convolutional layer

We now add the Conv1D layer to the network. This will have Filters of size 128 which is the dimensionality of the output space (i.e. the number of output filters in the convolution) and Kernel size of 5, specifies the length of the 1D convolution window. This window slides over the input matrix formed from embedding vectors and calculates local feature vector for each word. A transition matrix is generated for each filter. This filter comprises of layer of weights that are applied to parts of input matrix to obtain a single unit as output called transformed feature map, that acts as an input to next layer ReLU layer

4) ReLU layer

The ReLU layer acts as an activation function, which deals with the non-linearity in the model. Activates a node which is above zero. This layer creates a rectified feature map, this is fed to the next layer pooling layer.

5) Pooling layer

Down samples the input representation by taking the maximum value over the window defined by pool_size. Here we have taken pool_size=2. This layer reduces the dimensionality of the feature map and the LSTM layer is fed with all the composite features obtained from the above layers to judge Sarcasm.

6) LSTM layer

After adding the convolution neural network layers, we now add a LSTM(long short-term memory) unit to the network. This will have 100 outputs .

7) Fully connected layer

This is the output layer where , previous layers neurons are fully connected to every other neuron in the network. A softmax activation is added on top of Dense layer (fully connected layer), so that it gives the final single vector as output which is used for classification (sarcastic or non-sarcastic)

Training of network is performed by minimizing the binary cross-entropy error. For parameter optimization, we have used ADAM. Since the training in neural networks is an iterative process, the training won't just stop after it is done, So You have to specify the number of iterations needed for training of the model. These iterations are often called epochs(25 epochs are considered here). Another parameter, batch size(32 batch size is taken here) helps us to know the number of samples used in single epoch(one forward/backward pass)

D. CLASSIFICATION

The training sets are classified by using a script written in python language. The classification is done by referring the feature maps, which implies that the more features we extract, the better result we would get, but this increases the time to train the model. Fig.2. shows the working flow of our sarcasm classifier. In our model, CNN is used for feature extraction and classification is done by LSTM by adding a Softmax activation at the end. Thereby, we classified into two classes, namely sarcastic and non-sarcastic. Table-II shows some of the generated labels from our model.

Table-II: Some of the generated labels for Telugu news headlines

Telugu News Headlines	Label
న్యాయవాదిని చంపినట్లు ఆరోపణలు ఎదుర్కొంటున్న వ్యక్తి కొత్త న్యాయవాదిని ఎంచుకున్నాడు (English version: Man Accused of Killing Lawyer Receives a New Attorney)	Sarcastic
భద్రతా సమావేశం ప్రమాదంలో ముగుస్తుంది (English version: Safety meeting ends in accident)	Sarcastic
మణిపూర్‌లో నూతన ప్రభుత్వం ఏర్పాటు (English version: New government formed in Manipur)	Non-Sarcastic
ఇంగ్లండ్‌పై యువరాజ్ సింగ్ సరికొత్త బ్యాటింగ్ రికార్డ్(English version: Yuvraj Singh's latest batting record against England)	Non- Sarcastic

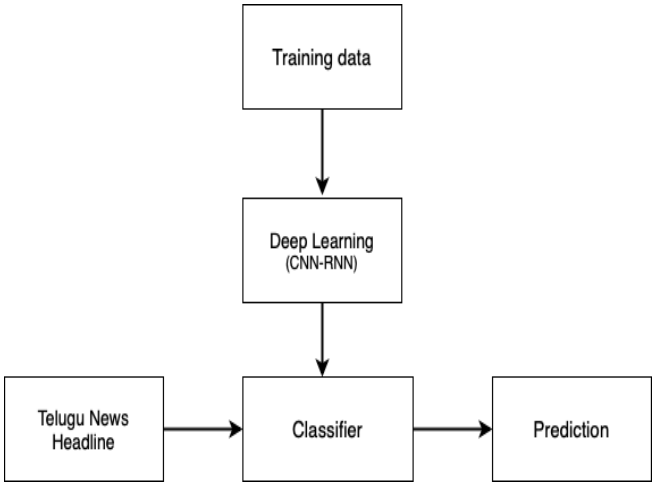

Fig. 2. Flow of sarcasm classifier

V. EXPERIMENTAL RESULTS

In our experiment we have taken a dataset of 4000 plus sarcastic and non-sarcastic sentences written in Telugu language and applied the deep learning techniques for feature extraction and classification. Table-III shows the values used in our work. The dataset is divided randomly into 70% for training and 30% for testing. The complete training is performed in CPU and keras python library as backend, TensorFlow library as front end.

We have observed that compared to baseline LSTM model, CNN-LSTM model has performed well. Table-IV shows the performance metrics (accuracy and loss) of the deep-learning models used.

Table-III: Hyperparameter values

Hyperparameters	Value
Dimension of Glove vectors	100
Number of convolutional layers	1
Number of convolutional filters	128
Window size	(5,5)
Activation function of convolutional layer	ReLu
Hidden units of LSTM	100
Batch Size	32
Regularization	Dropout operation
Dropout rate	0.4: word embedding
Learning rate	0.05: Glove model

Table-IV: comparison of the accuracies among models

MODEL	METRICS	
	ACCURACY	LOSS
CNN-LSTM	0.76	0.61
LSTM	0.62	1.73

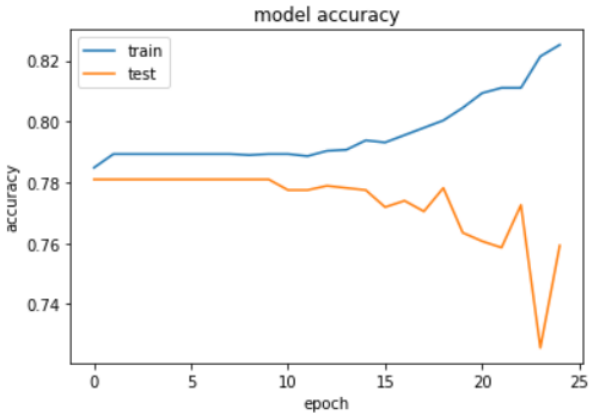

Fig. 3. Accuracy graph for proposed model CNN-LSTM

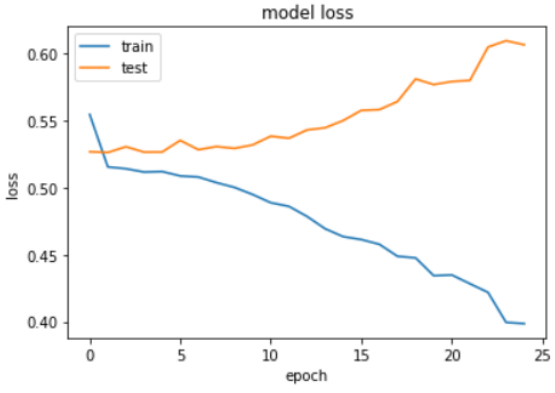

Fig. 4. Loss graph for proposed model CNN-LSTM

VI. CONCLUSION AND FUTURE WORK

In this work we have used convolutional neural network and LSTM(long short-term memory unit) on Telugu news headlines to find out sarcasm. The semantic word embeddings are taken as features to train the neural network. To detect sarcasm in the Telugu news headlines, first we should feed the headline to the model. Then we should apply our classifiers by using TensorFlow as frontend, keras as backend. Our model obtained an accuracy of 76%.

The future Sarcasm detection is that it can be performed using different data sets in different languages so as to provide local sarcasm detection, our proposed system can be improved by using pragmatics (emojis) in text as features of sarcasm. As we have used Telugu corpus as dataset it is very challenging task to train it with deep-learning techniques, so we need an improved corpus to various local languages.

REFERENCES

1. B. Liu, "Sentiment analysis and opinion mining," Synthesis Lectures on Human Language Technologies, May 2012 ,vol. 5, no. 1, pp. 1–167, 2012.
2. S. K. Bharti, K. S. Babu, and S. K. Jena, "Parsing-based sarcasm sentiment recognition in twitter data," in Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM) ACM, 2015, pp. 1373–1380.
3. M. Bouazizi and T. Ohtsuki, "A Pattern-Based Approach for Sarcasm Detection on Twitter," IEEE Access, vol. 4, pp. 5477–5488, 2016, doi: 10.1109/ACCESS.2016.2594194
4. Diana Maynard and Mark A Green wood, "Who cares about sarcastic tweets? investigating the impact of sarcasm on sentiment analysis", in Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14),2014, pp. 4238–4243
5. P. Dharwal , T. Choudhury, Rajat Mittal, Praveen Kumar "Automatic sarcasm detection using feature selection", in proceedings of 3rd International Conference on Applied and Theoretical computing and communication and Technology ,2017 IEEE ,doi : 10.1109/ICATCCT.2017.8389102
6. Ashwin Rajadesingan, Reza Zafarani , Huan Liu, "Sarcasm detection on twitter: A behavioral modeling approach", in Proceedings of the Eighth ACM International Conference on Web Search and Data Mining, ACM, 2015, pp. 97–106.
7. soujanya poria ,Erik cambria , Devamanyu Harzarika, prateek vij, "A Deeper look into sarcastic tweets using deep convolution neural networks", in Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, pp. 1601–1612, Osaka, Japan, December 11-17 2016.
8. Le Hoang Son et al. "Sarcasm Detection Using Soft Attention-Based Bidirectional Long Short-Term Memory Model with Convolution Network",vol.7,pp.23319-23328,2019IEEE,doi:10.1109/ACCESS.2019.2899260
9. Yessi Yunitasari , Aina Musdholifah , Anny Kartika Sari, "Sarcasm Detection For Sentiment Analysis in Indonesian Tweets", Indonesian Journal of Computing and Cybernetics Systems, Volume 13, Issue 1, 2019, pp.53-62
10. Shih-Kai Lin, Shu-Kai Hsieh, "Sarcasm Detection in Chinese Using a Crowd sourced Corpus", The 2016 Conference on Computational Linguistics and Speech Processing ROCLING 2016, pp. 299-310
11. Dana Al-Ghadhban , Eman Alnkhilan , Lamma Tatwany , Muna Alrazgan , "Arabic Sarcasm Detection in Twitter" , International Conference on Engineering & MIS ,2017 IEEE, doi : 10.1109/ICEMIS.2017.8272990
12. Santosh Kumar Bharti , Korra Sathya Babu, Rahul Raman, "Context-based Sarcasm Detection in Hindi Tweets" , 9th International Conference on Advances in Pattern Recognition (ICAPR-2017), doi : 10.1109/ICAPR.2017.8593198
13. J. Pennington, R. Socher, and C. Manning, ''Glove: Global vectors for word representation,'' in Proc. Conf. Empirical Methods Natural Lang.Process. (EMNLP), 2014, pp. 1532–1543.
14. Y. Kim. (2014). ''Convolutional neural networks for sentence classification.''[Online]. Available: https://arxiv.org/abs/1408.5882.

15. A. Ghosh and T. Veale, ‘‘Fracking sarcasm using neural network,’’ in Proc.7th Workshop Comput. Approaches Subjectivity, Sentiment Social Media Anal., 2016, pp. 161–169.
16. S. Hochreiter and J. Schmidhuber “Long short-term memory” Neural computation, 9(8):1735–1780, 1997.
17. Suneetha Eluri, Sumalatha Lingamgunta “ARPIT: Ambiguity Resolver for POS Tagging of Telugu, an Indian Language” published in i-manager Journal on Computer Science, Volume Issue 1, pp. 25-35, ISSN Print: 2347-2227, March-May 2019
18. Suneetha Eluri, Sumalatha Lingamgunta “A Statistical Method for Named Entity Recognition in Telugu, an Indian Language” published in International Journal of Recent Technology and Engineering (IJRTE): ISSN: 2277-3878, Volume -8 Issue-2, pp. 4211-4216, July 2019.

AUTHORS PROFILE

Dr. Suneetha Eluri is working as Assistant Professor in the Department of Computer Science and Engineering at Jawaharlal Nehru Technological University Kakinada, Andhra Pradesh, India. Her research interests are Natural Language Processing of regional languages with AI, Machine Learning and Deep Learning techniques. She is a Faculty champion of University Innovation Fellows programme at Stanford University. Currently she is working on NLP tasks and sentiment analysis of Telugu language. She has 16 years of academic experience. She has published a number of research papers in various reputed National and International Journals and Conferences. She has guided around 35 Post-graduates and 25 graduates of Computer Science and Engineering.

Naga Santosha Lahari Penmatsa is a Second year student at Jawaharlal Nehru Technological University, Kakinada, Andhra Pradesh, India. Pursuing her M.tech. in computer science and Engineering. She is currently working on her projects in Machine Learning and Deep Learning. This is her first paper on Deep Learning and its algorithms.