Grenze International Journal of Engineering and Technology, June Issue

Sarcasm Detection: A Survey on Sarcasm

Deepanshu[1], Devansh Suri[2], Gurmehar Singh Oberoi[3], Nandita Kalra[4], Dr. Nidhi Chandra[5] and Dr. Betty Paulraj[6]
[1-6]Amity University Uttar Pradesh Noida,India
Email: deepanshu@s.amity.edu,{ suridevansh02, gurmeharinj, nanditakalra2018}@gmail.com,{nsrivastava5, bpaulraj}@amity.edu

***Abstract*— In today's world there is need to develop a Sarcasm Detection model because of growth of social media. Due to rise in the usage of social media many people tend to write comments or posts that convey different meaning in one half and different meaning in other half, these types of sentences are known as sarcastic sentences. Now sarcasm detection is very useful in politics as well business because they tend to understand those text of the public and make changes in their way of handling their customers or followers. Sarcasm is frequently employed on social networks and microblogging platforms, where individuals use irony or criticism in a manner that can perplex even humans in determining whether the statement is sincere or not. Its figurative nature poses a significant hurdle for sentiment analysis. Although sarcasm carries an implicit negative sentiment, its outward expression often appears positive. The complexities surrounding sarcasm and the potential advantages of detecting sarcasm in the context of sentiment analysis have piqued interest in automatic sarcasm detection as a research endeavor. Automatic sarcasm detection encompasses computational techniques aimed at forecasting whether a provided text contains sarcasm. Sarcasm detection is a difficult task as it is even difficult for humans to understand a sarcastic text and it is even more difficult for a machine to understand a sarcastic text. It's already difficult for a machine to understand a normal text and store it in a sequential order hence creating a sarcastic text detector is difficult, and there is not much review work done on this. This review paper focuses on different techniques that has been used till now, but our main task is to focus on the best machine learning models that can be used for sarcasm detection.**

I. INTRODUCTION

NLP (Natural Language Processing) plays a vital role for sarcasm detection. Researchers are focused on NLP so that machines can understand human texts, chat reactions, and store them in a sequential order so that they can easily understand the message conveyed by the sentence. With gain in the use of social media such as Twitter, Facebook, WhatsApp, YouTube there is a need to develop models that can easily understand human texts and comments.

People tend to write texts, comments, and review such that one half of the sentence is conveying something else, and the other half is conveying something else. Sarcasm can be understood as conveying different meaning or the meaning opposite to what has been written in the text. For example, "Movie was so good that I slept till the end" now in this the first half convey that the movie was good, but the other half convey that the person slept while watching the movie that means that the movie was boring, and the person didn't like the movie. Sarcasm is difficult to recognize because of its implicit and explicit meanings. Textual sarcasm detection is very hard to detect without

Grenze ID: 01.GIJET.10.2.195

the facial expression, tone and nature of the person that wrote that context. That's why we need a way to find out a method or model to detect sarcasm using only textual data. Therefore, we required a well-developed NLP model for sarcasm detection.

There are some research papers on different models using different techniques for sarcasm detection but none of them discussed all of them together. "Reference [1][2][3]" For example, [1][2][3] are some of the research papers that talked about the models using different techniques like machine learning, deep learning, context-based model and all of them are still in the implementation phase. "Reference [1] For example, in this paper [1] a machine learning technique random forest with features of sentiment analysis gave the best results and without the features of sentiment analysis SVM model gave the best results. "Reference [3] And in this paper [3] LSTM and GRU models performed differently in different cases but gave major results with dot-product attention.

Some of the research papers reviewed all these models together but none of them concluded which model would be best for the detection of sarcasm. Sentiment analysis was also reviewed in this paper. Some deep learning approaches were also discussed but they didn't discuss models like transformers which are the best deep learning models available that is developed by Google.

Our aim of this is to identify or conduct a systematic literature review that is focused on identifying the gaps, challenges and best models available for the detection of sarcasm based on the development approach datasets and results. Moreover, future improvements and challenges will also be discussed in this article.

systems that can recognize sarcasm in textual content with accuracy. The main challenge at hand is developing a reliable and efficient model that can discern between sardonic and non-sarcastic comments across a variety of internet platforms, languages, and circumstances. Because sarcasm is so common in daily communication—especially on social media, where comprehension of the intended meaning is essential for precise sentiment analysis, opinion mining, and natural language understanding—this challenge is extremely important. The objective is to create a sarcasm detection system that can raise the standard and precision of text analysis instruments and raise awareness of user-generated material in the digital era.

III. Literature Review

A. Machine Learning Approach

Textual data ⟶ *Number*

For sarcasm detection, a number of machine learning algorithms have been investigated. Using supervised learning algorithms, such as Support Vector Machines (SVM), Naive Bayes, and Random Forests, is one well-liked strategy. These algorithms gain knowledge from data that has been labelled and where each instance has been classified as sarcastic or not. They then classify new instances using this training data. Approaches for supervised learning typically work best when there is a large amount of labelled data at hand.

Unsupervised learning is a different strategy that uses algorithms like topic modelling and clustering to find patterns and clusters in data without any prior labelling. Approaches to unsupervised learning have the advantage of not requiring labelled data, making them useful in situations where gathering labelled examples is difficult. They might, however, be less precise than supervised algorithms.

"Reference [1]" In order to automatically identify and categorize sarcastic statements in text, machine learning algorithms are essential for sarcasm detection. For this task, a number of machine learning algorithms have been investigated, each with advantages and disadvantages. Here is a detailed analysis of some of the machine learning techniques frequently employed for sarcasm detection:[1]

1. Support Vector Machines (SVM): SVM is used to effectively distinguish between these two classes. Finding a hyperplane that maximizes the margin between the sarcastic and non-sarcastic instances is the fundamental idea behind support vector machines (SVM). The hyperplane clearly divides the two classes by serving as a decision boundary. Since the SVM algorithm looks for the best hyperplane to differentiate sarcasm from non-sarcasm, it is especially reliable in this situation. This method is useful for detecting sarcasm because it makes use of SVM's advantages to identify the best boundary even in cases where the data may be complex or non-linear. SVM can efficiently classify text instances as sarcastic or non-sarcastic by learning from the labelled data and drawing well-informed conclusions about this boundary. This helps to advance the larger objective of automating the detection of sarcasm in textual content.

2. Naive Bayes: The probabilities that a text will fall into a sarcastic or non-sarcastic class are estimated using the probabilistic classification technique Naive Bayes. It achieves this classification by taking into account whether or not the text contains particular features. For sarcasm detection, the fundamental principle of Naive Bayes is the computation of conditional probabilities. After determining the class labels (sarcasm or non-sarcasm) and

calculating the likelihood of observing specific features, it applies the Bayes theorem to determine the likelihood that the text belongs in each class.
3. Random Forest: Random Forest creates a collection of decision trees for sarcasm detection, with each tree being built by taking into account a random subset of features and a random subset of training data. Because of the diversity that this unpredictability brings to the trees, the model is less prone to overfitting and is more resilient. Each decision tree in the ensemble independently determines whether the text is sarcastic or not when it comes time to make predictions. Combining the outcomes of each tree yields the final prediction, which is then decided upon by selecting the class with the highest average or majority vote.
Irony Detection in Sentiment Analysis
Understanding the intended meaning behind a statement is a prerequisite for both sarcasm and irony detection. Irony detection is frequently included as a subtask in sentiment analysis, which seeks to ascertain the sentiment or emotion expressed in a text. Rule-based approaches, which look for specific linguistic cues or patterns to identify ironic statements, and machine learning approaches, which train classifiers on labelled data to distinguish between ironic and non-ironic statements, are two examples of irony detection techniques.

B. Deep Learning Approach

In Machine learning we convert textual data to number due to which the sequential information of the text is lost, but deep learning uses sequential information. Feature Generation is done by deep learning model whereas in the machine learning approach we have to make it by ourselves.
Artificial Neural Networks (ANNs): Deep learning, inspired by the structure and function of the human brain, is based on artificial neural networks. ANNs comprise of layers of interconnected nodes (neurons) that read and convert data. These nodes are also called artificial neurons or perceptron's. These neurons are organized into three layered structural fashion comprising of input layer which is responsible for detecting and collection of raw data, hidden layers - these layers are placed in the middle and these intermediate layers are responsible for performing computations of high complexities. These comprise multiple layers which helps the deep learning of deep neural networks and output layer which is present in the final destination of the network and helps in concluding the deep neural learning through producing an output in the form of network predictions. Connections including weights and biases are responsible for the connection of perceptron's of one layer to the neurons of the very next layer. These are counted as the parameters of the network that justify the complexity of the network.
Activation functions take care to prevent the linearity in the required model. This is an important step in model designing as it approximates the relationships in the complex data gathered. These activation functions mainly consist of Sigmoid, ReLU (rectified linear unit) and Tanh. The main process is called the feedforward process where the data is allowed to flow through the input layer down the hidden layers and finally to the output layer, but here the connections, weights, biases and activation functions are processed upon each perceptron.
RNN (Recurrent Neural Network): It is better than ANN (Artificial Neural Network) because it has a special ability to give feedback to itself such that the sequence of the data is maintained. There is a hidden unit in RNN that provided the feedback to itself. Best for sequential data or time series data. Not able to retain context of long sentences. But it is not good for long sentences. There is problem of Vanishing/Exploding.
LSTM (Long Short-Term Memory): It's even better than RNN (Recurrent Neural Network) as it provides one more later or path so that it can even sequence the longer sentences to avoid the problem of Vanishing/Exploding.
There is also a different architecture inside LSTM that helps in communication between STM (Short Term Memory) and LSTM (Long Short-Term Memory).LSTM and STM both are paths that store the sequence of the data. Long short-term Memory is one level above RNN, it is able to retain context of long sentences as well hence it is majorly used. This falls under the category of recurrent neural network (RNN)in the deep learning . this algorithm is applied where the other RNN method fails due to lack of their competence of processing data with sequence as a priority and making it adaptable for natural language processing and speech recognitions etc. we would be incorporating this algorithm in our sarcasm detector and try to compare its efficiency with other methods of the machine learning concept. Some general concepts and factors of LSTM include:
Sequential data handling: LSTM are the methods which are used when the order of the data becomes significantly important and has an effect on the outcome of the process. So every data set that has a sequential aspect to it is treated with LSTM. Long term dependencies: LSTM's one of main features are that it is capable of detecting long term dependencies in the data set. This is done with the help of gating mechanisms. Gating mechanisms: The flow of data is controlled through these mechanisms within the network.
There are three types of gates.

Forget gate: the previous cell sends data to the current cell and forget gates decide which data is useful and which data needs to be forgotten.
Input gate: the input gate is responsible for deciding the new data that needs to be added to the current cell state.
Output gate: output gate is responsible for deciding which data needs to be left out as the exposed output of the cell structure.
Cell state: LSTM's have an isolated cell state that helps data flow across the time steps. Gating mechanisms update these cell states as per needed.
GRU/CNN: GRU is for text Generation. The Gated Recurrent Unit (GRU) is an alternative version of the recurrent neural network (RNN) architecture, aimed at overcoming the short-term memory limitation while offering a simpler structure in comparison to the Long Short-Term Memory (LSTM). In contrast to LSTM, GRU consolidates the input gate and forget gate into a single update gate, streamlining its design. Unlike LSTM, GRU doesn't maintain a separate cell state. A GRU unit comprises three key components: the update gate, reset gate, and the current memory content. These gates empower the GRU to selectively update and utilize information from previous time steps, enabling it to capture long-term dependencies within sequences.
CNN is mainly used for Image classification. Convolutional Neural Networks (CNNs) represent a potent category of deep learning models that find applications across diverse domains, such as object detection, speech recognition, computer vision, image classification, and bioinformatics. They have also proven effective in time series prediction tasks. CNNs are a type of feedforward neural network that harnesses convolutional structures to autonomously extract features from data. Unlike conventional approaches, CNNs possess the capability to learn and identify features from the data autonomously, eliminating the need for manual feature extraction by human experts. These networks are inspired by the workings of human visual perception and comprise key elements like the convolutional layer, pooling layer, and fully connected layer.

C. Context Based Models Approach

All the previous studies and research done on sarcasm detection shows or identifies the benefit of using contextual information about a author, comment, tweet etc. That helps build a more accurate result. All the previous models were extended using additional layers like LSTM, GRU, extra linear layer, dot product attention. Our discoveries demonstrate that consideration components related to relevant embeddings like Elmo can work on model execution without significantly expanding preparing time. Contrasted with our benchmark models, we saw gains with consideration while preparing somewhere around 50,000 models, implying that these more complex structures are just valuable with a generally huge dataset.

IV. METHODOLOGY

The detection of sarcasm using NLP requires a variety of advanced techniques and greatly benefits from machine learning algorithms. There are references to multiple NLP recipes in the above paragraph, and I will elaborate on how NLP can be used to detect sarcasm utilizing these techniques. Extraction of a noun phrase (recipe 1): Understanding the structural elements of a sentence depends on the use of noun phrases. Extraction of noun phrases can be used to identify discrepancies between the stated and intended meanings in sarcasm. Identifying phrases like "great job," for instance, in a sarcastic context can be useful. Text Similarity #2 (Recipe 2): The detection of sarcasm frequently relies on matching the text to previously observed patterns. By comparing them to recognized sarcastic templates or patterns in a dataset, text similarity calculations can assist in identifying sarcastic statements. The Following Are the Components of Speech Tagging: Identifying the intended use of words in a sentence, which is essential for understanding sarcasm, can be done with the aid of speech tagging. By classifying words as nouns, verbs, or adjectives, NLP models can provide context for sarcasm detection. Recognizing Named Entities (Recipe 4): Sarcasm frequently involves words, and NER can assist in identifying named words in text. The context provided by words like "Apple" or "Stanford University" can help you identify sarcasm. Classification of the Text (Recipe 6): Text classification techniques may be used to determine the meaning or intent of a statement. Classifying text into categories like "sarcastic" or "non-sarcastic" can help in sarcasm detection. Sentence Analysis (Recipe 7): Sentence analysis is a key component of sarcasm detection. Sarcasm can be distinguished by understanding the underlying sentiment and recognizing differences between the expressed and the intended sentiment. Word Sense Clarification (Recipe 8): Sarcasm frequently relies on wordplay and phrases with many meanings. Techniques for word sense disambiguation can aid in identifying whether a term is used in a sarcastic manner. Translation and language detection (Recipe 11): Sarcasm can be used in many different languages, and language detection can be used to determine the language of the text, making it possible to detect sarcasm in a

variety of languages. Sarcastic content in languages other than the source language can also be analysed with the use of translation.
In conclusion, NLP plays a pivotal role in sarcasm detection by enabling the analysis of text data, extracting relevant information, and identifying the nuances and contextual cues that signify sarcasm. The mentioned NLP techniques and recipes provide a foundation for building robust sarcasm detection systems, improving communication analysis, and enhancing sentiment understanding in various applications.

V. Dataset Collection

"Reference [3]" Self-Annotated Reddit Corpus (SARC) dataset is used for context-based sarcasm detection approach. It contains a corpus of 1.3 million sarcastic statements. A balanced version of the dataset is used, it contains 257,082 comments from variety of subreddits (forums). The SARD dataset is constructed in such a way that its half contains a sarcastic context and other half contains a non-sarcastic context.[3].
"Reference [16]" Reyes and Rosso (2014) collect a dataset of movie and book reviews, along with news articles marked with sarcasm and sentiment. In an earlier study, Reyes and Rosso (2012) garner 11,000 reviews of products with sarcastic expressions [16]
"Reference [6] Emojis and sentimental words are meant to convey the same idea when they appear together. Dictionary-based approaches and corpora are the two main ways to identify intuition. In English, conjunctions are usually used to join two statements [6]. To look into the attitude Dialect words are frequently employed. Text is divided into categories based on both its objective and subjective forms. Sentiment dialects are observed through the use of subjective phrases. To determine polarity, one uses weighted inverse document frequency (IDF) [6]. An equivalent situation is the Part of Speech (POS) tag, which may have words connected with it in the report.
"Reference [5]" Irony-expressing classification criteria serve as the primary role for feature classifications [5]. The Twitter and "Reference [4]" Amazon datasets are often used to train semi-supervised identification algorithms, enrich patterns, enrich punctuations, and teach patterns. Star ratings are also taken into account [4]."Reference [8]" The Twitter airline sentiment analysis was able to predict the polarity of tweets by using prominent machine learning methods like Naïve Bayes and Logistic Regression with feature extraction techniques like count vectorizer, TF-IDF, and word2Vec. It was found that the dataset responded effectively to the combination of these techniques. [8].
For the Machine Learning approach, a SemEval 2018-T3-train-taskA.txt dataset is used to find the best model from SVM, Random Forest, Naïve Bayes, Decision Tree.
Two of the datasets were used for sarcasm detection using a deep learning approach and those are news headlines and Reddit datasets. The news headline dataset was taken from two prime news websites, the Onion and the HuffPost. The sarcastic part was obtained from Onion whereas the non- sarcastic part was obtained from HuffPost. "Reference [10]" There were 23,976 non-sarcastic and 20287 sarcastic comments in the news headlines dataset. Of these, 5071 sarcastic and 5994 non-sarcastic remarks made up 20% of the testing data, with the remaining 80% of the data being utilized to train the model. [10]
"Reference [9]" A huge sample of satirical Reddit comments from online conversation was gathered to build this dataset [9]. The data is made up of both balanced and unbalanced versions in a ratio of 1:100. The corpus contains one million sarcastic statements as well as a few non-sarcastic remarks from the same source. 90% of the photographs in the Reddit dataset were used to train the model under discussion (449962-Sarcastic comments and 449,993-non-sarcastic comments). The model was evaluated on 10% of the photos (49995-Sarcastic comments and 49,999-non-sarcastic comments).

TABLE I. DATASETS AND CLASSIFIERS USED IN PREVIOUS WORK FOR SARCASM DETECTION USING DEEP LEARNING APPROACH.

Authors	**Datasets**	**Classifiers**	**Accuracy**
P. Shrikhande, V. Setty and A. Sahani [11]	26.7 k headlines with 11.7 k sarcastic headlines and 14.9 k non-sarcastic headlines	CNN-LSTM	86%
		BLSTM	86.13%
A. Kumar, S.R. Sangwan et al.	Sarc-H dataset with a total of 1004 tweets was built with 414 tweets labelled as	CNN-LSTM	91%

[12]	sarcastic and 590 labelled as non-sarcastic		
Porwal, Saurabh et al. [13]	Model was trained on 1000 sarcastic and non-sarcastic tweets and gave a result using 198 unknown tweets with 99 sarcastic and non-sarcastic each	RNN-LSTM	91%
D. Vinoth and P. Prabhavathy [14]	The dataset was collected from Twitter, consisting of 10,000 tweets with an equal distribution of sarcastic and non-sarcastic tweets	CNN	92%
D.K. Nayak and B.K. Bolla [15]	Dataset consisting of 26,700 headlines, where 11,700 headlines were sarcastic and 14,900 were non-sarcastic, collected from Orion and HuffPost	CNN	90%
Goel, P., Jain, R., Nayyar, A., Singhal, S., & Srivastava, M.	news headlines [10] dataset	CNN (without Word Embedding)	94.28%
		LSTM (without Word Embedding)	95.12%
		GRU (without Word Embedding)	94.47%
		CNN (with Word Embedding) GloVe	95.36%
		LSTM (with Word Embedding) GloVe	96.10%
		GRU (with Word Embedding) GloVe	95.64%
	Reddit	CNN (without Word Embedding)	71.48%,
		LSTM (without Word	71.82%

		Embedding)	
		GRU (without Word Embedding)	71.67%
		CNN (with Word Embedding) Word2Vec	73.19%,
		LSTM (with Word Embedding) Word2Vec	73.65%
		GRU (with Word Embedding) Word2Vec	73.34%

VI. Discussion

A. Approach

1.“Reference [1]” Machine learning Approach: In the machine learning approach, SVM, Naïve Bayes, Decision tree, Random Forest algorithms were used for the dataset SemEval2018-T3-train-taskA.txt. Initially SVM outperforms all the other algorithms with an accuracy of 64% but when used with the features of sentiment analysis, Random Forest gave an accuracy of 76% while SVM was lagging behind by just 2% which is 74% [1].
2.Deep learning Approach: For the news Headline dataset GloVe outperformed the other models that included word embeddings, achieving 95.36%, 96.10%, and 95.64%, respectively, for the CNN, LSTM, and GRU models. Without word embeddings, the accuracy obtained by the CNN, LSTM, and GRU models is clearly 94.28%, 95.10%, and 94.47%, respectively.
For the Reddit dataset the accuracy increases of the CNN, LSTM, and GRU models without word embeddings are 71.48%, 71.82%, and 71.67%, respectively. On the other hand, Word2Vec achieved accuracy of 73.19%, 73.65%, and 73.34%, respectively, when word embeddings were added.
3. Context based model Approach

B. Experiments

Self-Annotated Reddit Corpus (SARC) dataset is used for this context-based sarcasm detection. The SARD dataset is constructed in such a way that its half contains a sarcastic context and other half contains a non-sarcastic context. The input to our model is concatenation of two sentence embedding and the output that we get are either 0 or 1, where 0 represents the sentence is non sarcastic and 1 represents that the sentence is sarcastic.

C. Evaluation Method

For evaluation purpose we used F1 score. We also have to compute a macro – averaged F1 score to get better understanding of the result. In order to compute the averaged F1 score we first to compute the F1 score of both negative and positive classes. Macro averaged F1 score is the average of the F1 score.

D. Experimental Details

25 models with different combinations of hyperparameters were conducted. For each of the 25 models a single run was conducted with no dropout. For each of the challenger models 2 training runs were conducted in which there was one with dropout probability 0.2 and other with no dropout probability.
“Reference [3]” Since the dropout parameter and the number of LSTM/GRU layers were set for each model type, there were 36 potential permutations of the remaining hyperparameters for models without the additional linear layer. We were able to thoroughly investigate the bulk of our space of hyperparameters by training 25 models. On the Azure NV6 virtual machine, training runs of models without the additional linear layer took about 4 hours. For

the extra linear layer, the learning rate was fixed to 0.001 as it has been optimal for majority of the previous developed models and also reduced the number of combinations of hyperparameters from 144 to 48. The training time to run the model took approximately 6hrs on the Azure NV6 Virtual machine. Using the best and optimal hyperparameter of each of the 25 models, each model is trained for 5k, 10k, 25k, 50k, 100k examples or context and recorded there F1 score. [3]

VII. Conclusion

A. Conclusion To Machine Learning Based Models

"Reference [1]" From [1] it was found that SVM algorithm gives best accuracy and works well with huge dimensional area. This algorithm generally makes use of a subset of training points and finally uses limited memory space.
Results confirms that SVM provides the best results, but random forest outperforms the support vector machine algorithm to the accuracy of 76% using the features of sentiment analysis while using the sarcasm-detection.txt dataset whereas SVM with features of sentiment analysis gave an accuracy of 74%.[1]

B. Conclusion To Deep Learning Based Models

"Reference [2]" From this article [2] it's found out that According to the study's findings, the LSTM model with GloVe embeddings performed the best, with accuracy rates of 95.36%, 96.10%, and 95.64% for the CNN, LSTM, and GRU models, respectively. When utilizing word embeddings, the ensemble accuracies show a notable improvement over the results obtained without these embeddings. Word2Vec beat the other models on the Reddit dataset, with accuracy scores of 73.19%, 73.65%, and 73.34% for the CNN, LSTM, and GRU models, respectively. The weighted average of the ensemble model was 81.64%. The ensemble model without word embeddings had weighted averages of 80.27% and 98.09% for the two datasets, respectively.[2]

C. Conclusion to Context Based Models

For some examples the GRU baseline model was not able to identify the sentence as sarcastic or non-sarcastic, in one case the GRU baseline model was giving a result as non-sarcastic but when used GRU with attention then the sentence turned out to be sarcastic. With this we concluded that GRU with dot-product attention helps in focusing on the beginning sequence of the sentence more heavily that helped in identifying the sentence as sarcastic even though the sarcastic part was only in the beginning of the sentence not in the other part of the sentence.
'Reference [3]" F1 scores were used. If a sentence is sarcastic the result will be on else, it will be 0. It was found out that Deep learning will perform better with machine learning models and hence they must be used with machine learning models. In machine learning SVM performance was the highest and Random Forest with features of sentiment analysis provided the best results. Then we get to know that LSTM is the best model that can be used from deep learning as it can store the complete sentence with its sequence. Bidirectional LSTM provided the best results it gave an accuracy of 94.91. And in context-based models it was found that LSTM and GRU both performed better with dot product attention layer.[3]

Acknowledgement

We are grateful to all the individuals, Dr. Nidhi Chandra and Dr. Betty Paulraj for providing their assistance and guidance during the research process. We appreciate our institution, Amity University, as well for the opportunity and providing us with the necessary resources and supporting us throughout.

References

[1] Sentamilselvan, K., et al. "Detection on sarcasm using machine learning classifiers and rule based approach." IOP Conference Series: Materials Science and Engineering. Vol. 1055. No. 1. IOP Publishing, 2021.
[2] Goel, Priya, et al. "Sarcasm detection using deep learning and ensemble learning." Multimedia Tools and Applications 81.30 (2022): 43229-43252.
[3] Nicholas Benavides, Angelo Ramos et al" Context Based Model for Sarcasm Detection".
[4] M. Boia, B. Faltings, C.-C. Musat, and P. Pu,A :) Is worth a thousand words: How people attach sentiment to emoticons and words in tweets," in Proc. Int. Conf. Soc. Comput., Sep. 2013, pp. 345_350
[5] Mohd Suhairi Md Suhaimin, Mohd Hanafi Ahmad Hijazi, RaynerAlfred and Frans Coenen. "Natural Language Processing Based Features for Sarcasm Detection: An Investigation Using Bilingual Social Media Texts," in Proc. International Conference on Information Technology, 2017, pp. 703-709.

[6] A. Joshi, P. Bhattacharyya, and M. J. Carman. (Feb. 2016),"Automatic sarcasm detection: A survey" Available: https://arxiv.org/ abs/1602.03426
[7] Shuigui Huang, Wenwen Han, Xirong Que and Wendong Wang, “Polarity Identification of Sentiment Words based on Emoticons,” International Conference on Computational Intelligence and Security, 2017. pp 134–138.
[8] K. Sentamilselvan, D. Aneri, A. C. Athithiya and P. Kani Kumar International Journal of Engineering and Advanced Technology (IJEAT) ISSN: 2249 – 8958, Volume-9, Issue-3, February 2020
[9] Khodak, M., Saunshi, N., & Vodrahalli, K. (2018, May). A large self-annotated Corpus for sarcasm. In proceedings of the eleventh international conference on language resources and evaluation (LREC 2018).
[10] Misra, R., & Arora, P. (2019). Sarcasm detection using hybrid neural network. arXiv preprint arXiv:1908.07414.
[11] Shrikhande, P., Setty, V., & Sahani, A. (2020, November). Sarcasm detection in newspaper headlines. In 2020 IEEE 15th international conference on industrial and information systems (ICIIS) (pp. 483-487). IEEE.
[12] Kumar, A., Sangwan, S. R., Singh, A. K., & Wadhwa, G. (2023). Hybrid deep learning model for sarcasm detection in Indian indigenous language using word-emoji embeddings. ACM Transactions on Asian and Low-Resource Language Information Processing, 22(5), 1-20.
[13] Porwal, S.; Ostwal, G.; Phadtare, A.; Pandey, M.; Marathe, M.V. Sarcasm detection using recurrent neural network. In Proceedings of the 2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS), Madurai, India, 14–15 June 2018; pp. 746–748.
[14] Vinoth, D.; Prabhavathy, P. An intelligent machine learning-based sarcasm detection and classification model on social networks. J. Supercomput. 2022, 78, 10575–10594.
[15] Nayak, D.K.; Bolla, B.K. Efficient deep learning methods for sarcasm detection of news headlines. In Machine Learning and Autonomous Systems, Proceedings of ICMLAS 2021, Tamil Nadu, India, 24–25 September 2021; Springer: Singapore, 2022; pp. 371–382.
[16] Devarkar, Shreyas, et al. Sarcasm Detection & Sentimental Analysis of Tweets Using Support Vector Machine.Vol.6,2019,www.jetir.org/papers/JETIR1904107.pdf.