International Journal of Electrical and Computer Engineering (IJECE)
Vol. 14, No. 6, December 2024, pp. 6786~6796
ISSN: 2088-8708, DOI: 10.11591/ijece.v14i6.pp6786-6796

Deep learning-based attention models for sarcasm detection in text

Ganesh Chandrasekaran[1], Mandalapu Kalpana chowdary[2], Jyothi Chinna Babu[3], Ajmeera Kiran[4], Kotthuru Anil Kumar[5], Seifedine Kadry[6,7]
[1]Department of Computer and Communication Engineering, Sri Eshwar College of Engineering, Coimbatore, India
[2]Department of Computer Science and Engineering, MLR Institute of Technology, Hyderabad, India
[3]Department of Electronics and Communication Engineering, Annamacharya Institute of Technology and Sciences, Rajampet, India
[4]Department of Computer Science and Engineering, MLR Institute of Technology, Hyderabad, India
[5]Department of Electronics and Communication Engineering, Malla Reddy Engineering College, Hyderabad, India
[6]MEU Research Unit, Middle East University, Amman, Jordan
[7]Applied Science Research Center, Applied Science Private University, Amman, Jordan

Article Info

Article history:

Received Mar 21, 2024
Revised Jul 23, 2024
Accepted Aug 6, 2024

Keywords:

Attention networks
CNN+Bi-LSTM
Deep learning
Sarcasm
Text mining

ABSTRACT

Finding sarcastic statements has recently drawn a lot of curiosity in social media, mainly because sarcastic tweets may include favorable phrases that fill in unattractive or undesirable attributes. As the internet becomes increasingly ingrained in our daily lives, many multimedia information is being produced online. Much of the information recorded mostly on the internet is textual data. It is crucial to comprehend people's sentiments. However, sarcastic content will hinder the effectiveness of sentiment analysis systems. Correctly identifying sarcasm and correctly predicting people's motives are extremely important. Sarcasm is particularly hard to recognize, both by humans and by machines. We employ the deep bi-directional long-short memory (Bi-LSTM) and a hybrid architecture of the convolution neural network+Bi-LSTM (CNN+Bi-LSTM) with attention networks for identifying sarcastic remarks in a corpus. Using the SarcasmV2 dataset, we test the efficacy of deep learning methods BiLSTM, and CNN+BiLSTM with attention network) for the task of identifying text sarcasm. The suggested approach incorporating deep networks is consistent with various recent and advanced techniques for sarcasm detection. With attention processes, the improved CNN+Bi-LSTM model achieved an accuracy rate of 91.76%, which is a notable increase over earlier research.

Corresponding Author:

Jyothi Chinna Babu
Department of Electronics and Communication Engineering, Annamacharya Institute of Technology and Sciences
Rajampet 516126, AP, India
Email: jchinnababu@gmail.com

1. INTRODUCTION

Sharing information through the internet has become increasingly common over the decades because social media is the sole method for reaching out to young individuals and interpreted sentimentality as a person's attitude toward a singular objective. The existence of sarcastic content within those online data is impeding sentiment analysis, whether through a computer or a manual process. Identifying sarcastic information within such a large corpus of online information is time-consuming and difficult. As a result, there is a strong desire to create a human-computer interaction system that can categorize sarcastic text in a corpus. Given the advent of online platforms, researchers working in a broad range of fields, with machine vision, natural linguistic computation, and many others, now have much easier access to extensive and precise data collections [1], [2]. Interpretation of the accurate sentiment is extremely important when determining people's opinions and feelings regarding any item or brand because a prejudiced or incorrectly detected comment can confuse both people and computers themselves [3], [4]. The term "sarcasm" is used to express an individual’s negative feelings on a positive note. The usage of sarcastic statements in texts should be eliminated as these statements have a serious impact on sentiment analysis [5]. Past research related to sarcasm detection have relied on rule-based and other techniques that employ i) Lexical, ii) Pragmatic, and iii) Polarity variations, and so on.

Since sarcasm's original intent is frequently the exact opposite of what the user intends, many people strive to convey their opinions and use them to get attention from others. Affective computing and computational linguistics can strengthen decision support systems and customer service by uncovering customers'/users' personal preferences or removing items/product information that got negative customer reviews. Lack of familiarity with the situation's surroundings, and the particular subject could make it harder to recognize sarcastic remarks. Recognizing context is among the most difficult parts of content moderation.

Intelligent sarcasm recognition is considered a simple text categorization issue [6]. The use of semantics, machine learning, and rule-based strategies have all been suggested as sarcasm detection techniques in previous publications. However, they all have drawbacks. Semantic and syntactic characteristics are used by Nagwanshi and Madhavan [7] to identify sarcasm in literature. Nonetheless, they are frequently used text categorization components that are also useful for sarcasm detection. According to Shah and Shah [8], deep learning algorithms perform better at detecting sarcasm. In order to improve sarcasm detection, we present in this work a hybrid deep learning architecture that combines attention mechanisms with convolutional neural networks (CNN) and bi-directional long short-term memory (Bi-LSTM) networks. Whereas the Bi-LSTM component can comprehend long-range relationships and contextual information, the CNN component is made to pick up on local characteristics and patterns in the text. The model performs better overall as the attention mechanism helps it focus on important text passages. The attention network performs better than the state-of-the-art techniques regarding F1-score, accuracy, precision, and recall. Combining the advantages of CNNs with Bi-LSTMs, the hybrid architecture offers a more thorough comprehension of the text. By providing a unique method that incorporates techniques from deep learning to enhance the determination of sarcasm within textual data, this work contributes to the area. The findings show how the suggested approach may improve sentiment analysis tools, enabling more precise interpretations of user-generated material on social media. In conclusion, this work analyses pertinent literature, tackles the crucial issue of recognition of sarcasm in sentiment analysis, and presents a brand-new composite deep learning model.

2. RELATED WORKS

In prior studies, various researchers have commonly employed three strategies to understand sarcasm in text. The first is a lexicon-based strategy that combines dictionaries with features like unigrams, bigrams, n-grams, and part of speech (POS) characteristics. The second method to identify sarcasm uses machine or deep learning techniques. The third approach is the context-based approach that involves the Study of Relationships among words in a corpus to detect the presence of sarcasm. The various methods for sarcasm detection employed in existing literature is shown in the Figure 1. In their work, Bouazizi and Ohtsuki [9] used the POS feature with other pertinent variables to analyze tweets for sarcasm using a pattern-based approach. They may demonstrate that adding various elements will enhance the performance indicators for sarcasm detection. They used different classification techniques for sarcasm identification; among them, random forest (RF) performed the best. Nigam and Yadav [10] used a lexicon-based approach involving an external dictionary in their work for sarcasm detection. The limitation of this strategy is that it prevents the dictionary from including words from hybrid languages. Clews and Kuzma [11] used a lexicon-based approach to handle sarcastic information by matching a string with positive and interjection lexicons. They could minimize the time needed to collect and analyze the data for sarcastic identification. Zhang *et al.* [12] have used a neural network-based model to detect sarcasm from Tweets. Their model could avoid the drawbacks associated with manually extracted features like POS tags and sentiment lexicons. With the assistance of an appropriate word embedding approach, their approach resulted in improved outcomes. Several classifiers were employed by Prasad *et al.* [13] to determine the sarcastic content present in tweets by utilizing emoji and slang dictionaries. The best performance is obtained with the Gradient Boosting algorithm among several classifiers employed for sarcasm detection. The data pre-processing was done with techniques like stemming and lemmatization with the natural language processing tool kit (NLTK) tool kit. The relation between two speakers who share their views in Tweets is used by Bali and Singh [14] for sarcasm detection with logistic regression (LR). It involves understanding contextual information/relation between the participants or speakers in identifying sarcasm. A fuzzy-based approach was used by Meriem *et al.* [15] in their work on sarcasm detection to identify sarcastic tweets involving different data pre-processing methods. They could use the fuzzy-based approach as most of the tweets are fuzzy and suitable to be analyzed using fuzzy models.

Ahuja *et al.* [16] have tried a sarcasm detection approach using AdaBoost, gradient boost and naïve Bayes techniques on four datasets involving sarcastic content. Based on their results, it was concluded that the ML approach will yield better results compared to context-based methods and that performance can still be improved using deep learning methods. Govindan and Balakrishnan [17] considered hyperbolic tweets when using 12 different machine-learning approaches that are likewise feature-based to identify sarcasm. By utilizing several data pre-processing techniques, the performance of sarcasm detection is enhanced by extracting the hyperbole-related features using the Python library. Bharti *et al.* [18] used hyperbole tweets, which are identified by recognizing adverbs. Several rules were framed by Maynard and Greenwood [19] to determine the sarcastic content from the text data in their work. They could determine how sarcasm affects sentiment polarity and demonstrate how sarcasm detection helps the sentiment assessment model perform better. They could recognize the sarcastic content in tweets by assessing the sentiment linked with the hashtags, which was utilized to identify sarcasm in tweets.

Felbo *et al.* [20] developed a pre-trained algorithm that relied on an array of emojis and could recognize tweets that contained sarcasm. To reduce the over-fitting of the models, they have used word and phrase coverage-based pre-training techniques. Their work is built on LSTM architecture, and to attain cutting-edge performance, some layers of their model are frozen and unfrozen. Davidov *et al.* [21] used a quasi-technique with two phases to determine sarcasm in reviews related to some products. They have emphasized the data acquisition process by employing varying techniques before proceeding with the classification task.

Hazarika *et al.* [22] have retrieved contextual clues from social media material to identify sarcastic tweets to get beyond the restrictions of using manual characteristics for sarcasm identification. For classification, they have used CNN and with the help of contextual features, their model could outperform others in detecting sarcastic content. To detect sarcastic content, the content and context-based features are employed by Ren *et al.* [23] in their work with CNN. The contextual modeling of text is done first with help of the document modeling technique, and the syntactical features are extracted, fed to the CNN that identifies sarcasm.

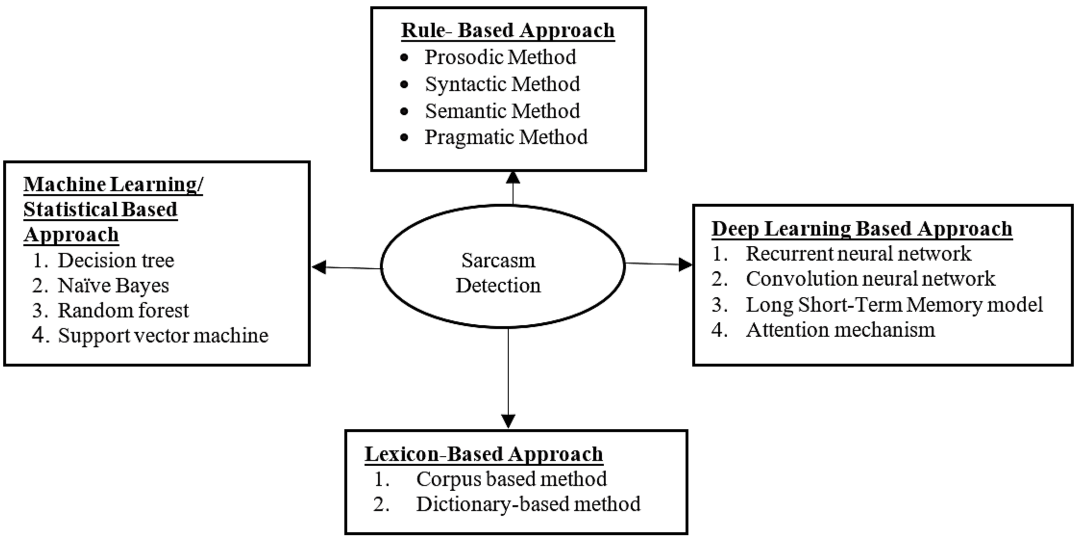

Figure 1. Strategies for sarcasm recognition

3. PROPOSED METHOD

This section contains a block diagram of the suggested text sarcasm detection scheme, which combines deep learning and a learning algorithm. Figure 2 depicts the anticipated analysis on sarcasm detection using textual information. The points that follow emphasize the various processes involved in understanding sarcasm.

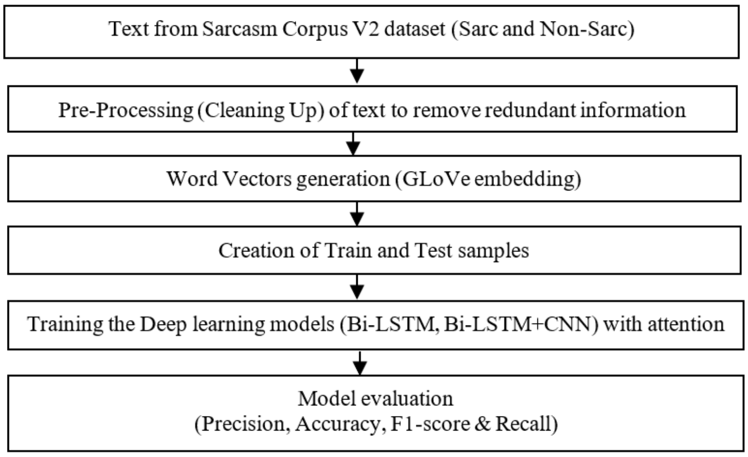

Figure 2. Process flow diagram of the suggested method for detecting sarcasm

3.1. Data acquisition

Many apostrophes, symbols, stopwords, and possibly other redundant elements in the dataset's raw text hinder sarcasm recognition. Before extracting word vectors, remove such data from the dataset. Tokenizing text with the NLTK package helps find and remove duplicate content. The data capture phase of the sarcasm classification starts with gathering text. There are a total of 9,386 instances of sarcasm in this dataset, of which 6,520 are generalized (GEN), 1,702 contain rhetorical inquiries (RQ), and 1,164 are hyperbolic (HYP).

3.2. Pre-processing of textual data

The dataset's raw text includes many apostrophes, symbols, stopwords, and perhaps other redundant aspects that do not help with sarcasm recognition. Consequently, before extracting word vectors, it is critical to eliminate that kind of information from the dataset. The NLTK package aids in tokenizing text, which divides large amounts of text into tiny tokens that assist in detecting and deleting duplicative content. The pre-processing step includes removing commonly encountered punctuation marks in addition to special characters (@, #, and so on). It is carried out with the aid of the Python "regular expression" package, which includes a variety of methods for scanning for and eliminating unwanted representations. The NLTK toolkit is then used to optimize the pre-processing outcomes using text normalization strategies, which include stemming, tokenization, and parts of speech marking. To prepare the model, the pre-processed information must be transformed into the input vectors in the following step.

3.3. Feature vectors generation with global vectors for word generation (GloVe) model

In contrast to other procedures for textual features that offer certain limitations, the global vectors for word generation (GloVe) word vectors procedure employed in the suggested model offers many merits. A GloVe is a cutting-edge approach that generates a low-dimensional tensor prototype without sacrificing word contextual resemblance. The first step entails creating a word dictionary from the words found in the dataset. This process generates a subspace of statements, each with a separate identity. In the following phase, the statistical detail of dataset words is crafted and concatenated to the feature space. In light of previous investigations on word embedding models, we could appraise the advantage of incorporating the GloVe model with the Bi-LSTM approach for better overall model performance. The data collection for the SarcasmCorpus V2 was divided into training and testing sets, and it contained 1,164 samples of hyperbolic sarcasm messages in addition to 6,520 instances of generic sarcasm and 1,702 samples of rhetorical dialogue sarcasm. The training set includes 7,508 samples, while the testing set includes 1,878 samples.

3.4. Training the attention models for sarcasm detection

The main goal of adopting deep learning design seems to be to avoid the limitations or obstructions that traditional machine learning designs have when providing the best solution. Algorithms for deep learning can automatically extract from a particular dataset without human intervention and are extremely efficient at capturing additional context. With machine learning and perhaps other algorithms, it is impossible to capture contextual information. The attention approach is employed to emphasize specific words that aid in detecting sarcasm and much less attention to insignificant terms.

3.5. Sarcasm detection with Bi-LSTM with attention model

The Bi-LSTM framework incorporates an attention system that enhances the sarcasm identification model's effectiveness. This method scores each key in the database against the vector space associated with the particular word in the input when a set of words is presented. To create attention output for the word being considered, the values are scaled in accordance with the attention weights and concentrate on the words that are pertinent to the request. A word's attention weight is calculated by considering the total amount of attention it has received throughout layers and afterward averaging the weights among respective heads inside this layer. The weights assigned to each word are then averaged over the entire phrase. The model gives the word a higher attention weight when classifying a sentence if it has a stronger highlight/importance. The block diagram in Figure 3 shows how the Bi-LSTM+Attention model is used to identify sarcasm.

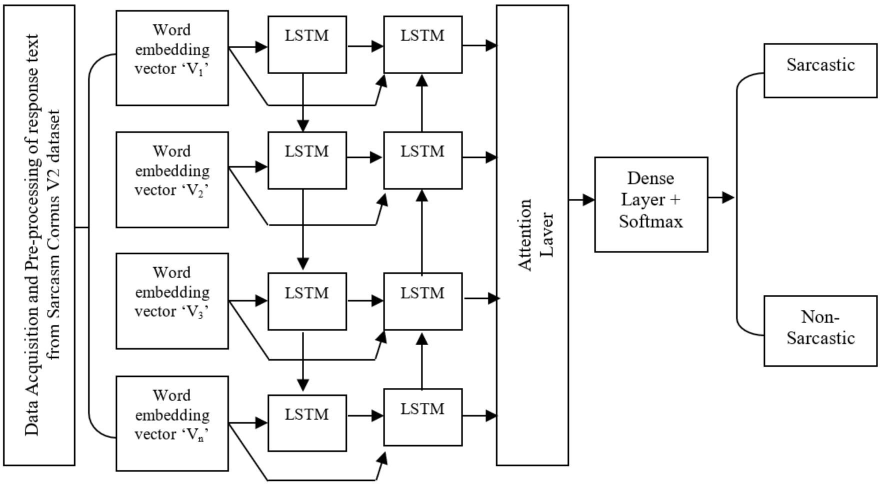

Figure 3. Bi-LSTM with attention text sarcasm detector block diagram

3.5.1. Architecture of the Bi-LSTM-based attention model

The following Table 1 details the different layers included in the Bi-LSTM model, featuring an attentive layer and several additional layers used for the planned study on sarcasm detection. The summary of the Bi-LSTM approach utilized to understand sarcasm is described in Table 1. The Keras library was used to build the model summary above, highlighting the several layers offered throughout the Bi-LSTM structure comprising the attention layer. Additional layers, especially dropout and thick layers holding certain units are added to increase classification performance. The word embedding vector obtained by the "GloVe" embedding approach is fed into the deep learning network. The idea of attention is accomplished by using layers such as permute, multiply, and lambda.

Table 1. Bi-LSTM model with attention – Keras summary

Layer (type)	Output Shape	Param #	Connected to
input 4 (InputLayer)	[(None, 50)]	0	[ ]
embedding 3 (Embedding)	(None, 50, 300)	7192500	['input_4[0][0]']
Bidirectional 2 (Bidirectional)	(None, 50, 128)	439296	['embedding_3[0][0]']
dense_10 (Dense)	(None, 50, 1)	129	['bidirectional_2[0][0]']
flatten_1 (Flatten)	(None, 50)	0	['dense_10[0][0]']
activation 1 (Activation)	(None, 50)	0	['flatten_1[0][0]']
repeat_vector_1 (RepeatVector)	(None, 128, 50)	0	['activation 1[0] [0]]
permute_1 (Permute)	(None, 50, 128	0	['repeat_vector_1[0][0]']
multiply 1 (Multiply)	(None, 50, 128)	0	['bidirectional_2[0][0]', 'permute_1[0][0]']
lambda_1 (Lambda)	(None, 128)	0	['multiply_1[0][0]']
dropout 6 (Dropout)	(None, 128)	0	['lambda_1[0][0]']
dense 11 (Dense)	(None, 256)	33024	['dropout_6[0][0]']
dense_12 (Dense)	(None, 128)	32896	['dense_11[0][0]']
dense_13 (Dense)	(None, 1)	129	['dense 12[0][0]']
Total params: 7,697,974			
Trainable params: 505,474			
Non-trainable params: 7,192,500			

3.6. Recognizing sarcasm combining CNN, Bi-LSTM, and attention model

To enhance the model's efficacy, the potential benefits of two major models, i.e., CNN and Bi-LSTM, are merged in this scheme for sarcasm detection. The convolution can retrieve the most characteristics from the input textual information, whereas the Bi-LSTM model can collect data both ways. The Bi-LSTM model's usage of various gates aids in disregarding undesired words/phrases. This method works best when dealing with a brief and continuous flow of information. The convolution layer extracts relevant attributes from text information, while the Bi-LSTM model extracts contextual information. The block diagram of the envisioned CNN+Bi-LSTM approach for sarcasm recognition is shown in Figure 4.

A single CNN model ignores the semantic and grammatical content of words in the context, and the challenge of vanishing gradients in classic recurrent neural networks (RNN) is easily handled using LSTM. After preprocessing the corpus, the text is encoded like a two-dimensional word vector matrix, and the local information features are recovered using CNN. The information is then sent to BiLSTM, which learns the link between words and phrases. Following the Bi-LSTM paradigm, the attention layer captures vital textual information while ignoring the uninteresting terms.

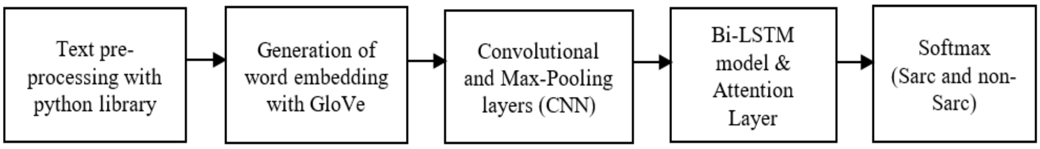

Figure 4. Text sarcasm recognition employing CNN+BiLSTM with attention

3.6.1. CNN+Bi-LSTM model with attention layer

The Table 2 provides the details of the several layers in the proposed network. Table 2 summarizes the Keras description of the CNN+Bi-LSTM network, with attention being used for sarcasm detection. The Keras library was used to construct the model summary above, which highlights the various levels made accessible throughout the CNN and Bi-LSTM framework that comprise the attention layer. To improve classification performance, further stages are added, especially layers of dropouts and thick layers holding particular units. The "GloVe" embedding approach yields a word embedding vector, which is injected into the deep learning network. The concept of attention is achieved using layers like permute, multiply, and lambda.

Table 2. Keras description of CNN+Bi-LSTM network for sarcasm detection with attention

Layer (type)	Output Shape	Param #	Connected to
input 4 (InputLayer)	[(None, 50)]	0	[ ]
embedding 3 (Embedding)	(None, 50, 300)	7192500	['input_4[0][0]']
dropout 7 (Dropout)	(None, 128)	0	['embedding_3[0][0]']
Conv1d (Conv1D)	(None, 46,32)	48032	['dropout_6[0][0]']
bidirectional 2 (Bidirectional)	(None, 50, 128)	439296	['embedding_3[0][0]']
dense_10 (Dense)	(None, 50, 1)	129	['bidirectional_2[0][0]']
flatten_1 (Flatten)	(None, 50)	0	['dense_10[0][0]']
activation 1 (Activation)	(None, 50)	0	['flatten_1[0][0]']
repeat_vector_1 (RepeatVector)	(None, 128, 50)	0	['activation 1[0] [0]]
permute_1 (Permute)	(None, 50, 128	0	['repeat_vector_1[0][0]']
multiply 1 (Multiply)	(None, 50, 128)	0	['bidirectional_2[0][0]', 'permute_1[0][0]']
lambda_1 (Lambda)	(None, 128)	0	['multiply_1[0][0]']
dropout 7 (Dropout)	(None, 128	0	['lambda_1[0][0]']
dense 11 (Dense)	(None, 256)	33024	['dropout_7[0][0]']
dense_12 (Dense)	(None, 128)	32896	['dense_11[0][0]']
dense_13 (Dense)	(None, 1)	129	['dense 12[0][0]']

Total params: 7,471,574
Trainable params: 279,074
Non-trainable params: 7,192,500

4. EXPERIMENTAL RESULTS AND DISCUSSIONS

4.1. Results

The source coding for this text sarcasm projection experiment was done in Python 3.5 with the aid of Google Collaboratory. It uses an ideal Tesla GPU featuring a 2.30 GHz clock speed and 16 GB of RAM. The Keras Python package is used to create the refined transfer learning model. The results were plotted using the MatPlotLib package, and the confusion matrix was created using the Scikit-Learn library. The Glove (*glove.6B.300d.txt*) word embedding is employed to generate word vectors before building up of the deep learning model.

4.1.1. Sarcasm V2 dataset

Sarcastic articles are included in this dataset (*https://nlds.soe.ucsc.edu/sarcasm2*). The Generic (Gen) category includes general sarcastic comment sequences. Hyperbole contains inflated, even absurd, deceptions that should not be taken literally. A rhetorical inquiry constitutes a query that is asked solely for effect and with no regard for the outcome. The features included within the SarcasmCorpus V2 are represented in Table 3.

There are 9,386 instances of sarcasm in this dataset, of which 6,520 are general (GEN), 1,702 constitute questions about rhetoric (RQ), and 1,164 are hyperbole (HYP). The input text data (Generic: 6,520 samples, rhetorical questions: 1,702 samples, hyperbole: 1,164 samples) are divided into the train (7,508 samples) and test (1,878 samples) with a vocabulary size of 2,3975 words to complete the sarcasm detection task. The word embeddings were initially stored in a dictionary with terms that shared the same key and value. The proposed networks were trained over 50 iterations using the 'Binary cross-entropy' approach. The confusion matrix describing the recommended strategy is shown in Figure 5.

Tables 4 and 5 display the performance indicators of the Bi-directional LSTM and CNN with Bi-directional LSTM models employing attention mechanism. According to the performance metrics, the CNN+Bi-LSTM classifier with attention layer produces the best accuracy (91.76%) for the specified input, text data, compared to the Bi-LSTM with attention (86.25%) algorithm for text sarcasm detection. The model's capacity to concentrate on important sections of the text was much enhanced by adding attention mechanisms, which increased performance metrics for all taken into account. Comparing our results to current sarcasm detection methods published in the literature, we find a significant improvement.

Table 3. SarcasmCorpus V2 details

Attribute/Feature Name	Description
Corpus	The corpus type: generic sarcasm (GEN), hyperbole (HYP), or rhetorical questions (RQ).
Label	The response utterance's class name - one of "*sarc*" (sarcastic) or "*notsarc*" (not-sarcastic)
ID	An identification number for the post. Pairs containing identical ID numbers in various datasets are unrelated.
Response text	The post's text is marked with sarcasm (i.e. this utterance is associated with the sarcasm label).

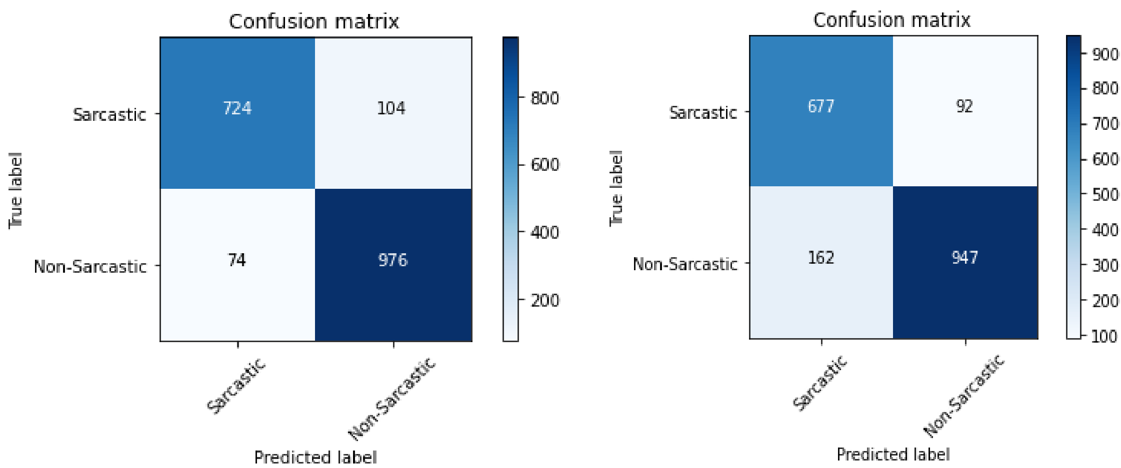

Figure 5. Deep learning models with attention–confusion matrix

Table 4. Bi-LSTM system performance measures with emphasis on sarcasm identification

Performance metrics	Values obtained
Sensitivity	80.72
Specificity	91.27
Precision	88.54
F1-Score	84.67
Accuracy	86.25

Table 5. Performance metrics of CNN+Bi-LSTM system with attention for recognizing sarcasm detection

Performance metrics	Values obtained
Sensitivity	90.27
Specificity	90.38
Precision	87.56
F1-Score	89.25
Accuracy	91.76

4.2. Discussion

The main conclusion from our findings is the efficacy of the suggested hybrid model (CNN+Bi-LSTM+Attention) in sarcasm detection in textual data. Specifically, the attention process improves the model's capacity to assess the relative relevance of various words and phrases, which raises detection accuracy. Table 6 demonstrates conclusively how much better our model performs than current models.

Our approach clearly outperforms alternative approaches when compared to previous study findings. This enhancement highlights the benefit of applying attention mechanisms in deep learning systems for sarcasm detection. Several artificial intelligence and deep learning methodologies have been used in a large number of published research projects to evaluate text sarcasm. We have contrasted our research with other relevant studies conducted using the SarcasmCorpus V2 collection. The following table compares our method for text sarcasm detection to various existing studies in Table 6.

Our study's thorough assessment of various model designs, which emphasizes the efficiency of attention processes, is one of its key strengths. But there are several restrictions to take into account: Computing complexity: not all experts may be able to train the suggested model due to its significant computing requirements. Data dependency: the caliber and volume of the sample dataset significantly impact the model's performance. The model's generalizability may be impacted by incomplete or skewed data. The results of the current investigation have significant ramifications for the creation of sentiment analysis algorithms that are more precise. These algorithms' ability to recognize sarcasm well allows them to provide sentiment ratings that are more accurate, which improves applications such as customer feedback analysis and social media monitoring. Potential areas of future research might include: i) Creating strategies to lower the model's computing demands without compromising its functionality and ii) To test the model's generalizability further, consider evaluating it on a wider variety of datasets, such as those from multiple social media sites and languages.

Table 6. Performance metrics of existing works versus the suggested for text sarcasm detection

Author	Dataset	Methodology used	Outcomes
González-Ibáñez *et al.* [24]	Twitter	Feature extraction using unigram and dictionary method to obtain Lexical and pragmatic feature. SVM classifier	Accuracy: 75.89% (polarity-based classification)
Lunando *et al.* [25]	Social media data of Indonesia	Negativity from the statements with their correlation with words. Traditional machine learning methods are used.	Accuracy: 53.1% (Naïve Bayes) Accuracy: 53.8% (Maximum Entropy) Accuracy: 54.1% (SVM)
Barbieri *et al.* [26]	Twitter	Features like words frequency, writing and speaking style, Impact of words, Structural formation, contextual sentiments and synonyms. Decision tree as classifier	Precision: 90%, Recall: 90% (Politics) Precision: 99%, Recall: 96% (Newspaper) Precision: 88%, Recall: 87% (Humor) Precision: 87%, Recall: 90% (Education)
Tungthamthiti *et al.* [27]	Twitter	n-gram features, pattern features and Punctuation features are extracted. Semi Supervised classification method.	Recall: 79% Precision: 78% F-measure: 79% Accuracy: 79.5%.
Bouazizi *et al.* [9]	Twitter online data	Pattern features and CNN	Accuracy: 83.10% Precision: 91.10%
Rajadesingan *et al.* [28]	Twitter	SCUBA approach to predict the previous tweets of users based on behavioral model.	Accuracy: 71.52 % (Emotion) Accuracy: 76.7% (Text)
Lukin and Walker [29]	Dialogue available in online	Pattern based feature extraction and Bootstrapping approach	Precision: 75% Recall: 62%
Gupta *et. al.* [30]	Twitter dataset	Word embedding is approach. Statistical method for classification.	Accuracy: 74.55%
Jena *et. al.* [31]	Reddit and Twitter dataset	Contextual network	F1-score: 75% (Twitter) F1-score: 66.31% (Reddit)
The proposed work (CNN+Bi-LSTM +attention)	SarcasmCorpus V 2 dataset	CNN, LSTM and Bi-LSTM with Glove word embedding	Accuracy: 91.76% Precision: 87.56% F1-Score – 89.25%

5. CONCLUSION

In this research, we have shown that several deep learning models-Glove word embeddings in particular-effectively detect sarcastic material within the SarcasmV2 dataset. Our models are based on preprocessing text with Python modules and converting phrases into geometric vectors using GloVe embeddings. The CNN coupled with a Bi-LSTM models performed better than the other examined designs. According to our research, the CNN+Bi-LSTM model incorporating attention mechanisms greatly increased accuracy to 91.76%, but the Bi-LSTM model only reached 86.25% accuracy. This significant improvement emphasizes how crucial it is to include attention processes to improve the model's capacity to concentrate on the most pertinent passages of the text, which will improve the detection of sarcasm.

The remarkable efficacy of the suggested approach in identifying sarcasm within textual data bears significant implications for the sentiment analysis domain. Sentiment modeling approaches, which are employed in many applications, including automated content moderation and customer feedback analysis may be made much more reliable by accurately detecting sarcasm. Our model advances the more general objective of creating natural language processing (NLP) systems that are more accurate by tackling the problem of sarcasm detection. This work demonstrates how cutting-edge deep learning architectures may help overcome the drawbacks of conventional sentiment analysis techniques. The supporting information of this manuscript is available in the following URL: *https://github.com/GaneshC86/Sarcasm-identification*.

REFERENCES

[1] R. K. Behera, D. Naik, S. K. Rath, and R. Dharavath, "Genetic algorithm-based community detection in large-scale social networks," *Neural Computing and Applications*, vol. 32, no. 13, pp. 9649–9665, Jul. 2020, doi: 10.1007/s00521-019-04487-0.

[2] A. Alsayat, "Improving sentiment analysis for social media applications using an ensemble deep learning language model," *Arabian Journal for Science and Engineering*, vol. 47, no. 2, pp. 2499–2511, Feb. 2022, doi: 10.1007/s13369-021-06227-w.

[3] H. Thakkar and D. Patel, "Approaches for sentiment analysis on twitter: a state-of-art study," *arXiv preprint arXiv:1512.01043*, 2015.

[4] G. Chandrasekaran and J. Hemanth, "Deep learning and TextBlob based sentiment analysis for coronavirus (COVID-19) using Twitter data," *International Journal on Artificial Intelligence Tools*, vol. 31, no. 1, 2022, doi: 10.1142/S0218213022500117.

[5] M. Sykora, S. Elayan, and T. W. Jackson, "A qualitative analysis of sarcasm, irony and related #hashtags on Twitter," *Big Data & Society*, vol. 7, no. 2, Jul. 2020, doi: 10.1177/2053951720972735.

[6] A. Ashwitha, G. Shruthi, H. R. Shruthi, M. Upadhyaya, A. P. Ray, and T. C. Manjunath, "Sarcasm detection in natural language processing," *Materials Today: Proceedings*, vol. 37, no. Part 2, pp. 3324–3331, 2020, doi: 10.1016/j.matpr.2020.09.124.

[7] P. Nagwanshi and C. E. V. Madhavan, "Sarcasm detection using sentiment and semantic features," in *Proceedings of the International Conference on Knowledge Discovery and Information Retrieval*, 2014, pp. 418–424, doi: 10.5220/0005153504180424.

[8] B. Shah and M. Shah, "A survey on machine learning and deep learning based approaches for sarcasm identification in social media," *Lecture Notes on Data Engineering and Communications Technologies*, vol. 52, pp. 247–259, 2021, doi: 10.1007/978-981-15-4474-3_29.

[9] M. Bouazizi and T. Otsuki, "A pattern-based approach for sarcasm detection on Twitter," *IEEE Access*, vol. 4, pp. 5477–5488, 2016, doi: 10.1109/ACCESS.2016.2594194.

[10] N. Nigam and D. Yadav, "Lexicon-based approach to sentiment analysis of tweets using R language," *Communications in Computer and Information Science*, vol. 905, pp. 154–164, 2018, doi: 10.1007/978-981-13-1810-8_16.

[11] P. Clews and J. Kuzma, "Rudimentary lexicon based method for sarcasm detection," *International Journal of Academic Research and Reflection*, vol. 5, no. 4, 2017.

[12] M. Zhang, Y. Zhang, and G. Fu, "Tweet sarcasm detection using deep neural network," *COLING 2016 - 26th International Conference on Computational Linguistics, Proceedings of COLING 2016: Technical Papers*, pp. 2449–2460, 2016.

[13] A. G. Prasad, S. Sanjana, S. M. Bhat, and B. S. Harish, "Sentiment analysis for sarcasm detection on streaming short text data," in *2017 2nd International Conference on Knowledge Engineering and Applications, ICKEA 2017*, 2017, vol. 2017-Janua, pp. 1–5, doi: 10.1109/ICKEA.2017.8169892.

[14] T. Bali and N. Singh, "Sarcasm detection: Building a contextual hierarchy," in *Proceedings of the Workshop on Computational Modeling of People's Opinions, Personality, and Emotions in Social Media (PEOPLES)*, 2016, pp. 119–127.

[15] A. Ben Meriem, L. Hlaoua, and L. Ben Romdhane, "A fuzzy approach for sarcasm detection in social networks," *Procedia Computer Science*, vol. 192, pp. 602–611, 2021, doi: 10.1016/j.procs.2021.08.062.

[16] R. Ahuja, S. Bansal, S. Prakash, K. Venkataraman, and A. Banga, "Comparative study of different sarcasm detection algorithms based on behavioral approach," *Procedia Computer Science*, vol. 143, pp. 411–418, 2018, doi: 10.1016/j.procs.2018.10.412.

[17] V. Govindan and V. Balakrishnan, "A machine learning approach in analysing the effect of hyperboles using negative sentiment tweets for sarcasm detection," *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 8, pp. 5110–5120, 2022, doi: 10.1016/j.jksuci.2022.01.008.

[18] S. K. Bharti, R. Naidu, and K. S. Babu, "Hyperbolic feature-based sarcasm detection in tweets: a machine learning approach," in *2017 14th IEEE India Council International Conference, INDICON 2017*, 2018, pp. 1–6, doi: 10.1109/INDICON.2017.8487712.

[19] D. Maynard and M. A. Greenwood, "Who cares about sarcastic tweets? Investigating the impact of sarcasm on sentiment analysis," in *Proceedings of the 9th International Conference on Language Resources and Evaluation, LREC 2014*, 2014, pp. 4238–4243.

[20] B. Felbo, A. Mislove, A. Søgaard, I. Rahwan, and S. Lehmann, "Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm," *arXiv preprint arXiv:1708.00524*, 2017.

[21] O. Tsur, D. Dmitry, and R. Ari, "Semi-supervised recognition of sarcastic sentences in twitter and Amazon," in *CoNLL 2010 - Fourteenth Conference on Computational Natural Language Learning, Proceedings of the Conference*, 2010, pp. 107–116.

[22] D. Hazarika, S. Poria, S. Gorantla, E. Cambria, R. Zimmermann, and R. Mihalcea, "CASCADE: contextual sarcasm detection in online discussion forums," in *COLING 2018 - 27th International Conference on Computational Linguistics, Proceedings*, 2018,

pp. 1837–1848.
[23] Y. Ren, D. Ji, and H. Ren, "Context-augmented convolutional neural networks for twitter sarcasm detection," *Neurocomputing*, vol. 308, pp. 1–7, 2018, doi: 10.1016/j.neucom.2018.03.047.
[24] R. González-Ibáñez, S. Muresan, and N. Wacholder, "Identifying sarcasm in Twitter: A closer look," in *ACL-HLT 2011 - Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 2011, vol. 2, pp. 581–586.
[25] E. Lunando and A. Purwarianti, "Indonesian social media sentiment analysis with sarcasm detection," in *2013 International Conference on Advanced Computer Science and Information Systems (ICACSIS)*, Sep. 2013, pp. 195–198, doi: 10.1109/ICACSIS.2013.6761575.
[26] F. Barbieri, H. Saggion, and F. Ronzano, "Modelling sarcasm in Twitter, a novel approach," in *Proceedings of the 5th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, 2014, pp. 50–58, doi: 10.3115/v1/W14-2609.
[27] P. Tungthamthiti, K. Shirai, and M. Mohd, "Recognition of sarcasm in microblogging based on sentiment analysis and coherence identification," *Journal of Natural Language Processing*, vol. 23, no. 5, pp. 383–405, 2016, doi: 10.5715/jnlp.23.383.
[28] A. Rajadesingan, R. Zafarani, and H. Liu, "Sarcasm detection on Twitter," in *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, Feb. 2015, pp. 97–106, doi: 10.1145/2684822.2685316.
[29] S. Lukin and M. Walker, "Really? well. Apparently bootstrapping improves the performance of sarcasm and nastiness classifiers for online dialogue," *arXiv preprint arXiv:1708.08572*, 2017.
[30] R. Gupta, J. Kumar, H. Agrawal, and Kunal, "A statistical approach for sarcasm detection using Twitter data," in *2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS)*, May 2020, pp. 633–638, doi: 10.1109/ICICCS48265.2020.9120917.
[31] A. Kumar Jena, A. Sinha, and R. Agarwal, "C-Net: contextual network for sarcasm detection," in *Proceedings of the Second Workshop on Figurative Language Processing*, 2020, pp. 61–66, doi: 10.18653/v1/2020.figlang-1.8.

BIOGRAPHIES OF AUTHORS

Ganesh Chandrasekaran hails from Salem district in the state of Tamil Nadu. He received his B.E degree in electronics and instrumentation engineering from Muthayammal Engineering College affiliated to Anna University, Chennai in the year 2008, his masters in applied electronics from Paavai Engineering College affiliated to Anna University, Chennai in the year 2013. He has 10 years of teaching experience and 2 years of industrial experience. His research interests include machine learning, deep learning, image processing and sentiment analysis. He has authored several research papers and presented papers in reputed international conferences. His research articles are cited by many international journals over the years. He can be contacted at email: csganesh86@gmail.com.

Mandalapu Kalpana Chowdary working as assistant professor, Computer Science and Engineering Department at MLR Institute of Technology, Hyderabad, Telangana, India. She received the B.Tech. degree in electronics and communication engineering from Chirala Engineering College, Chirala affiliated to JNTU, Kakinada in 2011. In 2013, received the MTech degree in electronics and communication engineering with VLSI Specialization from K L University, Vijayawada. In 2022, received the Ph.D. degree in electronics and communication engineering from Karunya University, Coimbatore, India. She has 7-year experience in teaching and 3.5 years in research, she published around 20 publications in international/national journals and conferences includes 5 SCI papers. 2 Patents in machine learning and deep learning. She is a life member of professional bodies like Indian Society for Technical Education (ISTE), IAENG. Research interest includes machine learning, deep learning, artificial intelligence, and image processing. She can be contacted at email: dr.kalpana@mlrinstitutions.ac.in.

Jyothi Chinna Babu has received his B.Tech. degree from JNTU, Hyderabad. M.Tech. degree in VLSI system design and Ph.D. in VLSI signal processing degrees from JNTUA, Ananthapuramu. Currently, he is working as associate professor in the Department of Electronics and Communication Engineering at Annamacharya Institute of Technology and Sciences-Autonomous, Rajampet, Kadapa, A.P, and India. He has published number of research papers in various international journals and various national and international conferences. He is a review member of various international journals. He was joined at AITS, Rajampet as an assistant professor in the Department of Electronics and Communication Engineering in the year 2008. He is having 15 years of Experience in Teaching UG and PG courses. He is a life member in ISTE and member in IACSIT, MIRED and CSTA. His current research interests are VLSI signal processing, microprocessor, embedded systems and communication, IoT, machine learning and signal processing. He can be contacted at email: jchinnababu@gmail.com.

Ajmeera Kiran working as assistant professor, Computer Science and Engineering Department at MLR Institute of Technology, Hyderabad, Telangana, India. He received the B.E degree in computer science and engineering from Vasavi College of Engineering affiliated to Osmania University, Hyderabad in 2012. In 2014, received the M.Tech. degree in computer science and engineering from Jawaharlal Nehru Technological University, Hyderabad. In 2021, received the Ph.D. degree in Faculty of Computer Science and Engineering from JNTUH, Hyderabad, Telangana, India. He has 3-year experience in teaching and 6.5 years in research, he published around 42 publications in international/national journals and conferences. 8 Patents in data mining and big data analytics. He also authored 6 books/books chapters by various international publishers. He was also a convener for various FDP's, coordinator for Student workshop's, Hackathon's. He is a life member of professional bodies like Indian Society for Technical Education (LMISTE), Member in IEEE. Research interest includes information security, network security, big data analytics and machine learning, artificial intelligence, data mining, internet of things. He can be contacted at email: kiranphd.jntuh@gmail.com.

Kotthuru Anil Kumar is currently working as an assistant professor in the Department of Electronics and Communication Engineering of Malla Reddy Engineering College Autonomous, Hyderabad, India. He has over 10 years of teaching experience. He received his B.Tech. degree in electronics and communication engineering from St. Mary's College of Engineering affiliated to JNTUH, Hyderabad in the year 2010, and his M.E. in applied electronics Sathyabama University, Chennai, India in the year 2012. His areas of research include digital signal processing, internet of things and digital electronics. He has published more than 5 articles that are indexed in Scopus journals and he also has presented research articles in National/International conferences. He can be contacted at email: kotthuru.anilkumar@gmail.com.

Seifedine Kadry has a bachelor's degree in 1999 from Lebanese University, MS degree in 2002 from Reims University (France) and EPFL (Lausanne), PhD in 2007 from Blaise Pascal University (France), HDR degree in 2017 from Rouen University (France). At present his research focuses on data science, education using technology, system prognostics, stochastic systems, and applied mathematics. He is an ABET program evaluator for computing, and ABET program evaluator for engineering tech. He can be contacted at email: skadry@gmail.com.