SN Computer Science (2024) 5:163
https://doi.org/10.1007/s42979-023-02506-5

ORIGINAL RESEARCH

N-Gram Based Sarcasm Detection for News and Social Media Text Using Hybrid Deep Learning Models

Chetana Thaokar[1,2] · Jitendra Kumar Rout[3] · Minakhi Rout[1] · Niranjan Kumar Ray[1]

Received: 6 May 2023 / Accepted: 20 November 2023

Abstract
Sarcasm is a type of communication that involves using words with meanings opposite to their literal definition to create humor or mock someone. This form of communication can be confusing as it often uses positive words to express negative feelings, making it difficult for people to understand the speaker's intention. Detecting sarcasm in text can be challenging as it changes the polarity of the sentence and the difference between the words used and the way it is spoken. As a result, sarcasm detection in news, comments, or tweets on social media poses a challenge for researchers. In this research, various word-level features have been studied to detect sarcasm from three benchmark datasets, which include the creation of an N-gram probability dictionary, negation words, and PoS tags. Different machine learning and hybrid deep learning models have been examined and compared with handcrafted features and also with word embedding features. The results demonstrate a maximum accuracy of 87% using manual features with deep learning models and up to 92% accuracy using RF classifiers.

Keywords N-gram probability · Negation feature · CNN · LSTM · Word embedding

Introduction

With the advancement in technology and the accessibility of social media at a lower cost, the number of social media users has grown exponentially. This presents a plethora of opportunities for users to share and comment in the form of text expressions. People consume various forms of information, such as news, blogs, articles, comments, reviews, and react to them. At times, news or articles are circulated for entertainment, to mock, or to defame the people. People who use social media often use sarcastic text to express their feelings or their views. Sarcastic text has a direct effect on sentiment analysis as well. For example, "Your mobile is fantastic as it has less battery life". This is a sarcastic sentence with a positive word, fantastic, but with a negative feeling. Therefore, sarcasm detection also improves sentiment analysis. It is, therefore, important for users to understand the thin line between text that is positive or deliberately negative. Different terms are used to justify this context, such as irony, satire, or sarcasm. Irony is a fun contradiction; satire is a type of criticism used to taunt a person that is not apparent to the person; while sarcasm creates fun at a person and is easily understood.

Detecting sarcasm is one of the most challenging tasks for humans as well as in natural language processing (NLP). As it involves determining whether the true meaning of a word is intended in a given context. Irony is often expressed as a form of sarcastic speech, where the speaker conveys an implied message to criticize or taunt a particular person. Tone plays a crucial role in communication. However, with many communication sites,

This article is part of the topical collection "Innovation in Smart Things: A Systems, Security, and AI Perspective" guest edited by Niranjan K Ray, Prasanth Yanambaka and Rakesh Balabantaray.

✉ Jitendra Kumar Rout
jitu2rout@gmail.com

Chetana Thaokar
chetana.thaokar@gmail.com

Minakhi Rout
minakhi.rout@gmail.com

Niranjan Kumar Ray
rayniranjan@gmail.com

1 School of Computer Engineering, KIIT Deemed to be University, Bhubaneshwar, Odisha, India

2 Ramdeobaba College of Engineering and Management, Nagpur, Maharashtra, India

3 Department of Computer Science and Engineering, National Institute of Technology, Raipur, Chattisgarh, India

Published online: 08 January 2024

such as Instagram, Facebook, and Twitter communication is restricted to text characters, making it challenging to determine the true meaning of a sentence.

Recognizing sarcasm helps prevent misinterpreting sentences that mean the opposite of their literal meaning. It is also applicable to other areas of NLP, such as machine translation, information retrieval, information mining, and knowledge acquisition. Currently, numerous studies focus on recognizing sarcasm or irony. Different methods have been proposed using statistical modeling, sentiment analysis, pattern recognition, and supervised or unsupervised machine learning. However, recognizing sarcasm involves more than an intelligent system that requires the development of psychological and linguistic forms of language understanding. According to [1, 2], the use of satire and sarcasm is studied to derive definitions and demonstrate some characteristics of satire. Studies show that satire and sarcasm arise from opposing intentions expressed by the opposite meaning of a sarcastic statement.

The motivation behind sarcasm detection is rooted in the need to understand and interpret the nuanced nature of human communication. Detecting sarcasm in text is challenging, because it relies heavily on contextual cues, tone of voice, and shared background knowledge. There are several reasons why sarcasm detection has garnered attention in research, as it contributes to improving various NLP applications, such as sentiment analysis, dialog systems, and social media analytics. In sentiment analysis, sarcasm can significantly impact the sentiment expressed in a statement. Distinguishing between sarcastic and literal statements can lead to more accurate sentiment analysis, allowing a better understanding of people's opinions and emotions in online conversations, product reviews, or social media discussions. Sarcasm is prevalent on online communication platforms like social media, forums, and chat applications. Accurately detecting sarcasm can help in understanding the true meaning behind user-generated content, reducing misunderstandings, and improving the overall quality of online interactions.

Sarcasm can be used as a tool to spread misinformation or misleading information. By detecting sarcasm in news and social media text, we can identify instances where individuals are using sarcasm to mask false or misleading claims. This can aid in the detection and prevention of the spread of misinformation. Overall, the main aim behind sarcasm detection is to enhance the capabilities of natural language understanding systems, improve sentiment analysis, and reduce misinterpretation of user intent.

In this paper, we present a concatenation of handcrafted text features combined with a hybrid deep neural network for sarcasm detection and its comparison with word embedding features. The main contribution of this paper can be summarized as follows:

- We used different text features, such as N-gram probability, important stop words, negation words, POS of words, interjections, and punctuation symbols.
- One hot encoding feature vector is created and this is given as input to the CNN structure.
- Used a hybrid model of CNN LSTM and CNN BiLSTM for finding significant features in the text to enhance the results.
- We compared and reproduced the results on the hybrid model using word embedding, pretrained word embedding, and context-aware techniques. Our model performed marginally well as compared to word embedding features.

The rest of our paper is outlined as follows: Sect. Literature Survey revisits the existing work done in this domain, Sect. Proposed Methodology elaborates on the methodology used, Sect. Experimental Setup discusses experimental setup and parameters used, and Sect. Results and Discussion and Conclusion and Future Directions discuss results and future directions in this given task.

Literature Survey

The detection of sarcasm through automated means is a relatively new area of research that has gained popularity in recent years. Previous works in this field can generally be categorized into two types: content-based and context-based models. Researchers have focused on identifying lexical and pragmatic features that can aid in detecting sarcasm in a given sentence [3–5], and various approaches have been developed and tested in the literature, showing promising results in detecting interesting signals for identifying sarcasm. Sarcasm often involves presenting cues, such as interjections, intensifiers, punctuation, and markers of non-veridicality and hyperbole to emphasize failed expectations [3]. These cues like "yay!" "great!" "wow" or "ohhh" are commonly found in sarcastic text, written for product reviews or general text.

A CNN-based deep neural network was introduced by Amir et al. [6]; authors suggested an approach of automatically learning and utilizing user embeddings, in addition to lexical signals, for identifying sarcasm. Their method avoided the need for extensive feature engineering or data scraping, as user embeddings could be generated using only the text from their previous posts. The results of their experiments demonstrated that their model surpassed a leading approach that relied on a large set of meticulously designed features. A study by Zhang et al. [7] compared the efficacy of continuous automatic features with discrete manual characteristics to investigate the application of neural networks for sarcasm detection in tweets. The researchers used a pooling neural network to automatically extract contextual data from earlier tweets and a bidirectional gated recurrent neural network to extract both syntactic and semantic information from tweets. The research discovered that, compared to discrete manual characteristics, neural features yielded better sarcasm detection accuracy's with different error distributions. Additionally, the researchers discovered that contextual tweet features for sarcasm identification in the neural environment were just as successful as discrete models.

Ghosh and Veale [8] designed a neural architecture to detect satire in a timely and contextual manner. They demonstrated that knowledge of the speaker's mood at the time of production can significantly improve detection accuracy. The study focused on detecting sarcasm on Twitter and found that not only the context of the topic but also the mood expressed by the speaker in the previous tweets leading up to a new post were useful cues for detecting satire. This work has the potential to not only explore satire in the text but also the satirical mental state. CASCADE, a hybrid approach of context- and content-based modeling, was proposed by Cambria et al. [9] to detect irony in online discussions on social networks. CASCADE aims to extract contextual information from the discourse of the discussion thread and uses user embedding to encode the user's stylistic and personality traits. When combined with content-based feature extractors like CNNs, it showed a significant improvement in classification performance across a large Reddit corpus. Pelser and Murrell deployed [10], a 56-layer deep network with dense connectivity to model isolated speech and extract richer features from it. They compared their approach with recent modern architectures and showed competitive results using only local text features. They also presented a case study demonstrating that their approach correctly classifies additional uses of the word simple satire, which the CNN standard classifies poorly.

Kumar et al. [11], proposed an attention-based bidirectional LSTM to automatically detect sarcasm. They introduced a multi-head attention mechanism to improve the performance of the BiLSTM, which outperformed the feature-rich SVM model used in the previous studies that produced models with lexical, semantic, and practical features. The test results showed that the proposed approach improved the detection of sarcastic comments in a given corpus. In a different study, Razali et al. [12] focused on detecting sarcasm in tweets by combining deep learning features with manual contextual features. They extracted a feature set from a CNN architecture and combined it with handcrafted features that were tailored for the unique task of detecting sarcasm. The authors ranked the feature combinations using several machine learning techniques and found that logistic regression was the best classification algorithm for this task. The results were positive in terms of accuracy, precision, recall, and F1 measure. In a similar kind of work, Bharti et al.[13] proposed a deep learning-based approach for multimodal satire detection by combining text and audio features. They found that the combined model produced significantly better results than the individual models. A systematic review of the literature on automated satire detection was presented in [14], showing that multimodal approaches and transformer-based architectures have become increasingly popular in recent years. The paper also critiques the previous work and suggests directions for future research in this area.

A semi-supervised learning approach was employed to classify sarcastic sentences on social media platforms and for online product reviews [15]. The approach comprises two main modules: semi-supervised pattern acquisition and classification algorithms. It filters a set of high-frequency words (HFWs) and content words (CWs) as a pattern for a sarcastic sentence. Then, it constructs a single feature vector for each pattern by calculating the feature value for each pattern based on its similarity to the other extracted pattern. Finally, the approach applies a K-nearest neighbor (kNN)-like method along with the feature vector to classify the sentences. This strategy does not emphasize semantic analysis but rather the importance of using the clustering of HFWs and CWs in a sentence. It is based on the idea that verbal irony requires a violation of expectations and felicity conditions for discourse acts. Therefore, if we observe both conflicting eagerness and violations of felicity conditions within a context, we can identify a sarcastic context.

Currently, unsupervised learning methods for detecting sarcasm are still in the early stages of development, with most algorithms being clustering-based and more suitable for pattern recognition. Researchers are working toward creating unsupervised models to avoid the limitations and difficulties associated with labeling datasets in supervised learning approaches, such as the time and labor-intensive process. In 2016, Nozza et al. [16] presented an unsupervised framework for detecting irony called the TopicIrony model (TIM), which builds upon probabilistic topic models, specifically the Latent Dirichlet Allocation (LDA) model, originally developed for sentiment analysis. TIM is designed to model irony toward various topics in a domain-independent manner. Mukherjee and Bala [17] also explored both supervised and unsupervised learning environments using the Naive Bayes algorithm and the Fuzzy C-means (FCM) clustering algorithm, respectively. Transfer learning approaches, which leverage pre-trained models and adapt them to new domains, have gained attention due to the difficulty of annotating data and the importance of context in capturing figurative language phenomena. A hybrid neural architecture called Recurrent CNN RoBERTA (RCNN-RoBERTa) was developed, which combines a recurrent convolutional neural network with the RoBERTa architecture to improve the detection of sarcasm [18].

Oprea and Magdy [19] developed neural models to extract author context, which they described as the embedded representation of a user's previous tweets. They used two different sets of tweets, one of which was human-categorized for sarcasm and the other of which was automatically labeled. In the author's proposed architecture, exclusive models make predictions based entirely on the user's past behavior rather than the currently classified tweet. On the other hand, inclusive models considered both previous user activity and the most recent tweet. Because of their rising profile in recent years, multimodal techniques also need mention. As part of the first multimodal technique, photos were incorporated into the sarcasm detection information. The authors compiled information from tweets, Instagram posts, and weblogs. After that, they used a support vector machine (SVM) method and a deep learning method to identify sarcasm. They utilized extracted NLP and visual semantic features for the SVM method. The DL strategy combined two separate networks to make the prediction: an NLP network and a visual network. Results showed that when visual information was integrated, performance improved for the Instagram set but had no effect on Twitter. Nevertheless, text features contributed little to the deep learning method's efficiency. In contrast, they developed a hierarchical fusion model that integrated not just one but three feature representations: picture, image attribute, and text. Textual characteristics, visual characteristics, and visual attributes were all considered separate modalities by the paper's writers. The paper's proposed hierarchical fusion model began with the extraction of image and attribute features before moving on to the extraction of text features using attribute features and a bidirectional long short-term memory (BiLSTM) network. The model then merged the information from the three modalities into a single feature vector for prediction purposes after reconstructing the features. For training and evaluation purposes, the authors used a Twitter-based multimodal dataset.

In addition to this, a multimodal strategy for text preprocessing that was based on BERT was proposed in [20]. The research was carried out using data from Twitter, which included both text and images. The model incorporated three different elements: text, a hashtag, and an image. Both intermodality attention, which refers to the relationship between picture and text, and intramodality attention, which refers to attention paid within the text, were used in the model. One fact that stands out as particularly intriguing is that the solutions with the highest scores in the SemEval 2020 competition made use of ensemble methods and/or applied data augmentation. As a result, semi-supervised techniques, when combined with transformer-based architectures, have the potential to achieve higher outcomes compared to other methods and should be favored moving forward.

The vast majority of publications make use of datasets of varying sizes. If we want to develop solutions that are more effective, then future research should concentrate on constructing datasets and dealing with them in a way that takes into account the context of the problem. Augmentation techniques can also be used to extend the dataset as discussed in [21]. Researchers are now showing interest in multimodal research methods, and it is recommended that further datasets similar to MUStARD [14], which combines several distinct kinds of data, be established. Based on the above discussion, Table 1 shows the overview of work done till now. To summarize, researchers have been exploring various approaches to automatic sarcasm identification, including unsupervised models such as TIMg, supervised models such as SVM, RF, Naive Bayes, CNN, LSTM, BiLSTM, and multimodal models that incorporate visual and image features.

Some recent approaches have also incorporated transfer learning and ensemble methods for improved performance. However, the effectiveness of these models is highly dependent on the size and context of the datasets used for training and evaluation, and there is a need for more comprehensive and diverse datasets that include a variety of data types. Overall, further research in this area has the potential to improve the accuracy and applicability of sarcasm detection in real-world settings.

Proposed Methodology

This paper aims to explore the classification task [22] using machine learning and deep learning models for sarcasm detection in news articles and social media text. It mainly uses handcrafted word-level features in both types of classification tasks. The proposed work flow in sarcastic text detection is shown in Fig. 1. As mentioned in the figure, input text goes through three steps: text filtering, feature extraction, and classification.

Sarcasm Filter

The text used for analysis is first filtered, so that only meaningful text can be further used. So first, complete text is converted to lowercase so as to maintain uniformity throughout. Unwanted symbols, such as numbers and new lines, are removed. Punctuation symbols are removed, but not all punctuation symbols used in writing text are removed, as some symbols like comma, !, and ? can help to understand the sentiment or intention of the text. Further tokenization is carried out, which breaks the sentences into words. The data set from which each piece of data is taken is in the form of sentences; after tokenization, these tokens are useful for finding patterns or features in it. Not all the tokens

Table 1 Overview of related works

Author	Dataset	Method	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Amir et al. [6]	Twitter	Cue-CNN	80.00	–	–	–
Zhang et al. [7]	Twitter	Glove	90.74	–	–	–
Ghosh et al. [8]	Twitter	CNN	–	73.33	71.70	72.50
Erik et al. [9]	Twitter	LSTM	92.80	–	–	–
Pelser and Murrell [10]	News Headlines	dweNet GLoVe	88.67	–	–	–
Kumar et al. [11]	Reddit	MHA-BiLSTM	–	72.63	83.03	77.48
Razali et al. [12]	Twitter	LR	94.00	95.00	94.00	94.00
Bharti et al. [13]	MUStARD	SVM	–	67.50	66.66	67.08
Baroiu and Matu [14]	MUStARD	SVM	–	61.00	45.50	48.70
Davidov et al. [15]	Twitter	KNN	–	91.20	–	82.70
Nozza et al. [16]	Twitter	Unsupervised	–	84.14	81.74	82.92
Mukherjee and Bala [17]	AMT	SVM	–	80.10	73.60	75.20
Potamias et al. [18]	Reddit	RCNN	79.00	78.00	78.00	78.00
Oprea and Magdy [19]	Rilof	Embedding	–	–	–	82.70

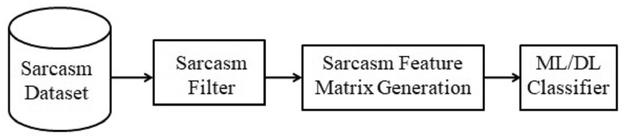

Fig. 1 Sarcasm detection workflow

Table 2 Example of text filters

Filter used	Sample text	Corrected text
Commonly used abbreviation LOL	Missing You! LOL	Missing You! Lots of love
Contraction	Im, going...	I am going...
Emoticon	: –) happy	Smile happy
	: –(I hate you	Frown I hate you

are meaningful for analysis; tokens that are less useful are removed and this process is called stop word removal. We have prepared a list of stop words that do not contribute to the meaning of the sentences, such as am, the, from, on, etc., The words but, that, not, never, etc. can help to understand the intention whether it is normal or negative with sarcasm, hence not removed; e.g., "What a luxurious hotel but we are soo rich that can't pay its charges". This sentence can be considered as sarcastic way of expressing the feeling. As news text is written by professionals, impurities in the form of slang are null, but social media text might contain slang and other impurities, hence being removed from the text. We identified impure words in social media text and corrected them. Therefore, further filters were applied, such as abbreviation, contraction, HTML tag removal, and emoticon sense expansion [23]. Sample examples are shown in Table 2.

For the above filters, a separate dictionary of words and its expansion were created to understand the meaning of shortcuts used while writing the text. The last step used in filtering was converting a word to its root word using the process of lemmatization. Lemmatization usually refers to the morphological analysis of words with the aim of removing inflectional endings. Lemma not only cuts out inflections but also relies on lexical knowledge bases like WordNet to get the correct base word forms; e.g., in English, the suffixes, such as *organize, organizing, and organization*, can be mapped to organize.

Feature Matrix Generation

Feature extraction is a measure task in determining the outcome of any machine learning or deep learning task. The quality of classification, both qualitatively and quantitatively, depends on the features selected. This paper focuses on extracting features from news headlines and social media that can be categorized into various types. It creates one hot encoding vector for each word present in the sentence based on four features; such as the POS tag of a word, which emphasizes finding verb terms and adjectives in the sentences, which are more common terms in identifying the intention of the user. Stop word feature, Negation words, which change the intention, and lastly, the N-gram feature based on the target label The vector size is 15 values for each word. This feature helps to find pattern of the text that contains sarcasm and identifies commonly used patterns for the same. This approach has the potential to detect sarcasm, with some limitations. The main problem that exists with the current technique is its inability to perform well in varied domains. Techniques that are used on sentiments perform well on data from specific domains that contain sentiment-related words. However, it varies as per the domain, whereas sarcasm depends not only on sentiment but also on the type of words used. [24] has shown how to reduce the feature dimension and retain only useful features if the feature matrix is very large. As in our case, features are few, so no dimension reduction techniques were used. Figure 2 shows the feature generation method.

POS Tag Encoding

There are a lot of words in English that have multiple parts of speech (POS), i.e., a word can be either a noun, verb, adjective, or adverb. The part of speech of each word varies, and it depends on the sense of the text. E.g., part of the speech of a word could be a noun in the first sentence and an adjective in another sentence. With the change in POS of a particular word, the absolute sense of the sentence gets changed. Therefore, the conclusion is that the absolute sense of a sentence depends on the POS of the words in the sentence.

E.g., Back: Noun, "his back was nicely tanned"

E.g., Back: Adverb, "he moved back".

From the above illustrations, it is clear that the sense of the word changes according to context; hence, the POS tag of a word becomes necessary to identify whom, who, and where these remarks are made. This feature comprises nine values, one for each tag; e.g., noun, pronoun, verb, adverb, adjective, determiner, conjunction, modal, particle. Based on the tag of the word nine, values are set or reset. Different variants of each tag are mapped to the base tag, such as the NN, NNS, NNP, and NNPS tags, which are mapped to the noun. The same is done for all other tags. Along with this, interjections are identified by making use of the tag

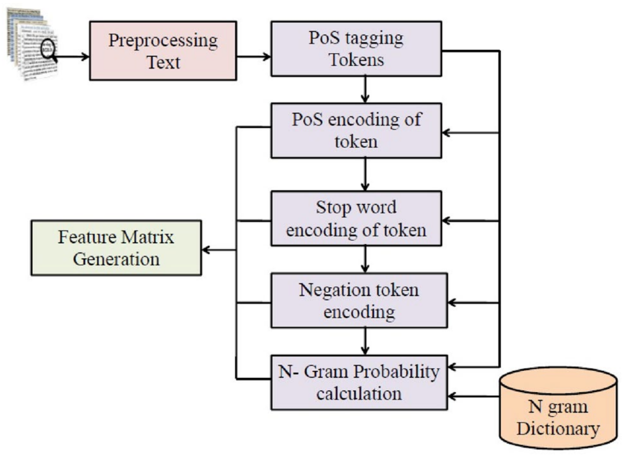

Fig. 2 Feature generation of sarcasm detection

determiner, which helps in understanding the emotion of the user.

StopWord Encoding

Frequent-appearing words that do not significantly contribute to the meaning of the sentence are stop words. Stop words can change the meaning of a sentence, and therefore, this feature is important in the feature matrix. Not all stop words are useful. Stopwords that are used as conjunctions are important and hence not removed. Therefore, a customized dictionary was created instead of the NLTK list. This feature is a single-value feature. If the token of text or sentence is present in the list of stop words, then the feature value is set to 1; otherwise, it is zero.

Negation Token Encoding

These words or tokens may completely change the meaning of the sentence. A negative word list is created, and if any word matches the list, it is set to 1, otherwise 0. It is an indication that it contains negation words, which reverse the meaning of the original sentence. E.g., "Obama asks Biden not to stand so close". "Obama asks Biden to stand so close." From this illustration, we can conclude that negative expressions change the polarity of the sentence and can also help in sarcasm detection. The first example is sarcastic, whereas the second is not.

N-Gram Probability Feature

It generates a dictionary of words with their corresponding sarcastic and non-sarcastic probability values from the training data set for unigram and bigram tokens. Figure 3 shows the step for dictionary creation. This dictionary contains the normalized probability of the tokens appearing in sarcastic and non-sarcastic sentences. A minimum threshold is used for finding the probability of the gram occurrence in the sentence. The trial-and-error threshold used was three occurrences. Algorithm 1 describes the formulation for N-gram dictionary creation, which calculates Ps and Pns, and uses the value of the total number of sarcastic and non-sarcastic sentences and the threshold value. Table 3 shows the sample result of the algorithm.

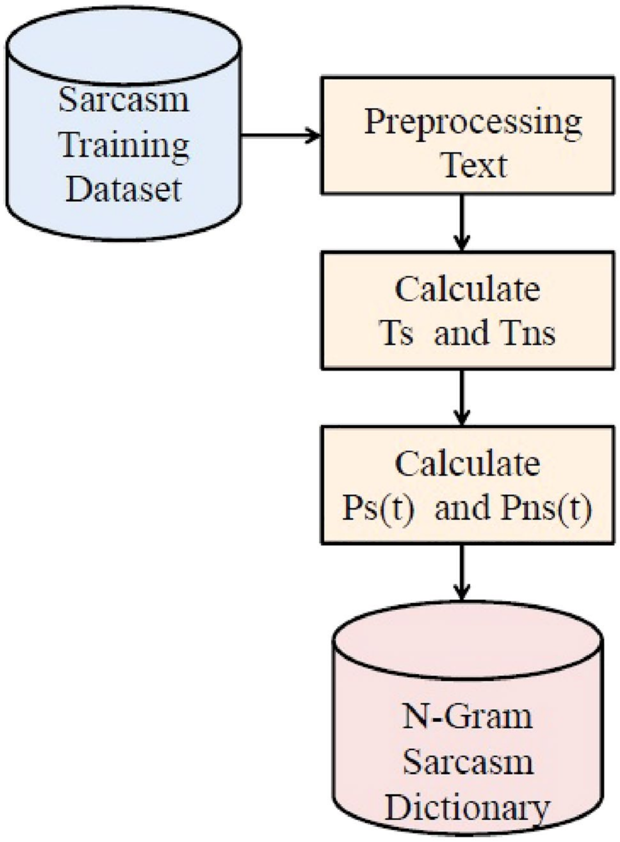

Fig. 3 N-gram Dictionary Creation

Table 3 Results of N-gram

N-gram	Token	Ps	Pns
1	Secret	0.46	0.53
1	Service	0.59	0.40
1	Walking	0.5	0.5
1	Dead	0.0	1.0
2	Walking dead	0.0	1.0
2	Secret service	0.72	0.27
2	Global warming	0.75	0.25
2	As much	0.42	0.57

Experimental Setup

In this paper, we have demonstrated the workings of our model on three different datasets that were downloaded from Kaggle. The three datasets used are news headlines, Reddit, and Twitter. Only the Reddit dataset has millions of texts, whereas the other two have thousands. The experiments were done on Google Colab. The dataset used had only two labels: sarcastic and non-sarcastic text. Out of three datasets, two have news contents collected from news sites or social media, and one is collected from Twitter. The Sarcasm-based News dataset is taken from Kaggle in JSON format. It contains a total of 28620 records, out of which 14985 are labeled as non-sarcastic and 13635 as sarcastic. This dataset cannot be called completely imbalanced, and hence, no efforts were made to convert it to balance it. The dataset consists of three fields: binary label, news headline text, and the URL of the article from where it was taken. The records of the headlines are collected from the two famous websites, The Onion and HuffPost. The onion is popular for its sarcastic content and is taken from the News in Brief and News in Photos sections. Whereas non-sarcastic contents are taken from the HuffPost website, In general, the dataset contains American headlines the most.

Since the headlines are written by professionals, there are fewer chances of spelling mistakes or the use of slang language, as it is openly used on social media platforms. For our convenience, we have deleted the URL field of the dataset, as analysis is only done on news text.Reddit is an American website for content collection, news creation, and social discussion forums where text, images, and videos are posted and comments are given by users as votes. The Reddit sarcasm dataset was created by [25] by the annotation of 1.3 million texts from the official website of Reddit. The Twitter dataset comprises nearly 60000 tweets, where sarcastic labels are very few. The author [20] collected the tweets with the hashtags sarcasm and #not and then labeled them. As it is unbalanced, we worked with 10% of the tweets with an 80:20 ratio. The

Algorithm 1 Formulate N-gram probability for Bigram and Unigram

```
1: for tokens in sentence do
2:     if $Ns$ >=th OR $Nns$ >=th then
3:         $Ps \leftarrow$ Ns / $Ns$+Nns
4:         $Pns \leftarrow$ Nns / $Ns$+Nns
5:     end if
6: end for
```

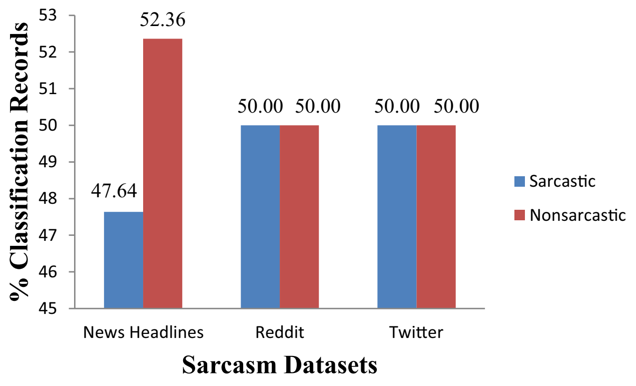

Fig. 4 Sarcasm dataset classification

complete distribution of text in the dataset is shown in Fig. 4, and sample text can be seen in Table 4.

Therefore, all the filters and feature extraction techniques were used on three datasets, and further classifiers were used. For the purpose of comparison, we have used five machine learning classifiers, viz., Decision Tree, Random Forest, AdaBoost, Gradient Boosting, Naive Bayes, and Deep Neural Architecture. For the machine learning classifier, a handcrafted feature matrix was given as an input, and the default setting was used. On the other hand, two kinds of models were made for deep neural architecture: first, handcrafted features were extracted and then fed into the deep learning architecture; and second, Keras Word Embedding was used to make a 200-word numeric vector. The deep learning architecture used is CNN (Conv1D), LSTM, BiLSTM, and hybrid models, as shown in Fig. 5.

A pipeline is used that transforms input text into a numeric word vector, and then, DL models are applied for the extraction of word-level features used for binary classification. CNN is useful in extracting both types of features, such as temporal and spatial, whereas LSTM and BiLSTM are good at extracting temporal features in sequential mode. Our neural architecture model comprises 2 layers of convolution with 64 and 32 units, with 64 batch sizes, a learning rate of 0.02, one layer of global maxpooling, and a dropout of 0.2. Optimization function as Adam and Loss function as binary cross entropy. Applied to this setup for various batch sizes and epochs, accuracy increased when batch size was 64 and early stopping criteria were used. In LSTM and BiLSTM, one layer of 64 units and a dropout layer of 0.2 are used. whereas for input as word embedding setup, epoch 10 was set and other parameters used were as above mentioned. To improvise on results, a hybrid model was examined as CNN LSTM and CNN BiLSTM in pipeline. And also in the concatenation-based approach, in which two model results are applied to one dense layer. However, in our setup, concatenation of two models did not work as required and resulted in nearly 65% accuracy. The experiments were carried out using an 80:20 ratio of data with the hyperparameters described above. To verify the results, different sets of training records were used

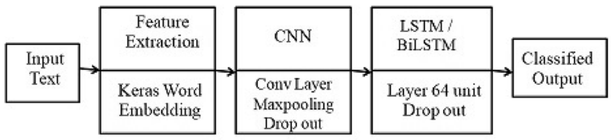

Fig. 5 Hybrid model architecture

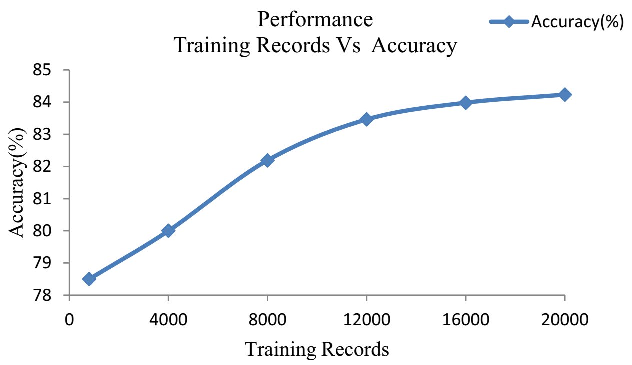

Fig. 6 Performance of model with different training records

Table 4 Sample text examples from the datasets

News headlines/social media text	Sarcastic / Not sarcastic
Report: U.S. Death rates from drugs, suicide, and alcohol have greatly increased, but not in a cool rock and roll way	1
Salma hayek rips donald trump: 'he has never done anything for America	0
Love working on holidays	1
Trump reassures struggling farmers he has never seen one of them and cannot be sure they even exist	0
Obama asks biden not to stand so close	1
No no no, Trump does not lie, he just invents new truths.	1

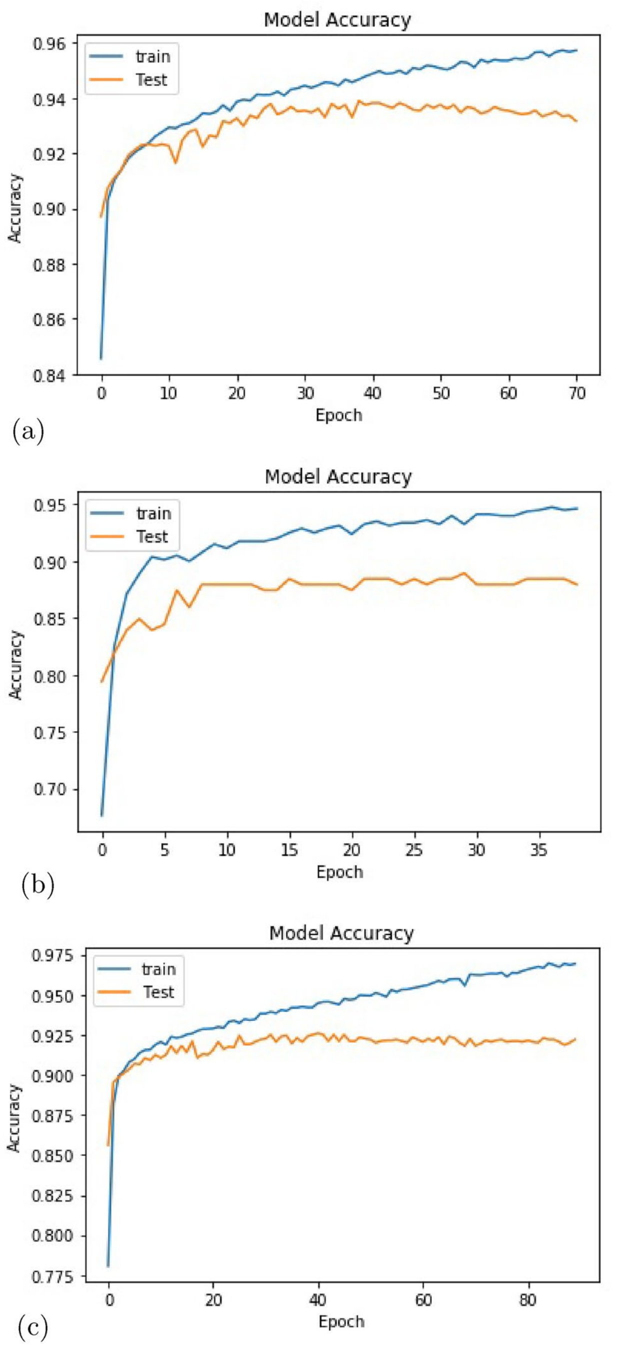

Fig. 7 Accuracy of best model on dataset: **a** News Headlines; **b** Reddit; **c** Twitter

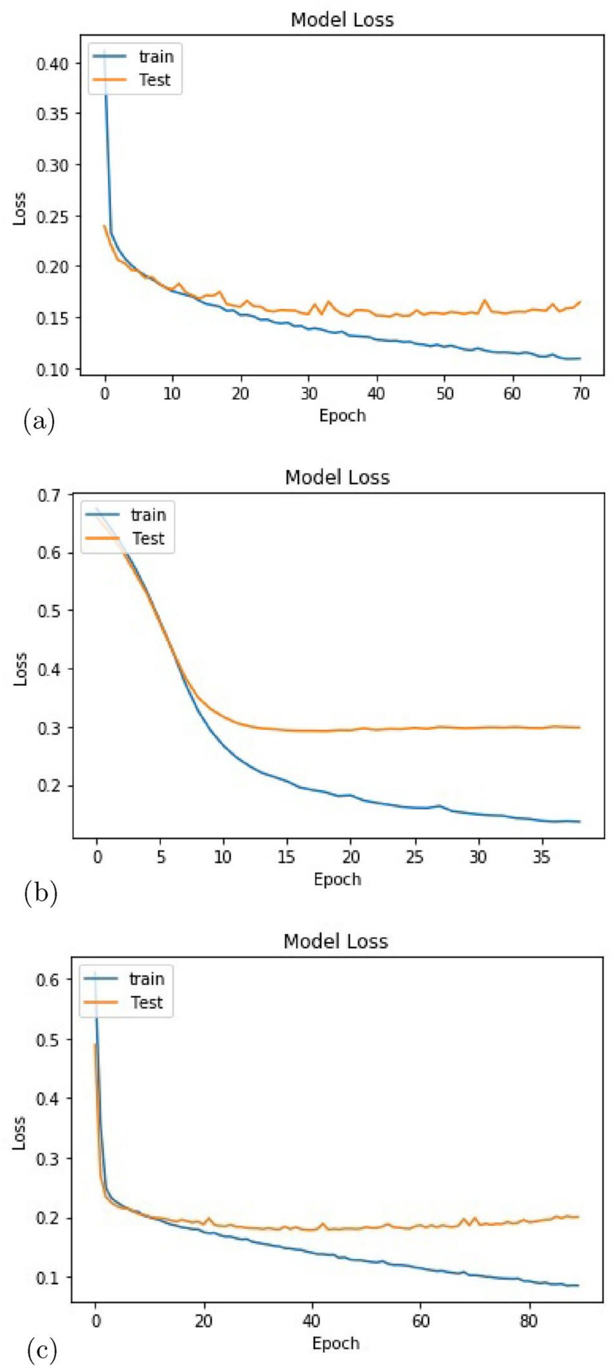

Fig. 8 Loss function of best model on dataset: **a** News Headlines; **b** Reddit; **c** Twitter

and observed. As training records increase, accuracy also increases, as can be seen from Fig. 6.

Figure 7 shows a cross-epoch comparison of training and testing accuracy of the best-performing model on the News, Reddit, and Twitter datasets using handcrafted features. It is evident from the graph that accuracy improves with epochs. Testing accuracy after a specific epoch remains the same, which is illustrated by the early stopping criteria. Figure 8 shows the training and testing losses for the best-performing models for three datasets. From

Table 5 Performance of ML algorithm

Classifier	News headlines	Twitter	Reddit
Decision Tree	80.56	80.72	82.06
Random Forest	82.45	83.78	92.91
Ada Boost	76.89	72.65	75.48
Gradient Boosting	78.87	71.05	77.05
Naïve Bayes	79.56	67.34	66.89

the figure, it is clear that overfitting is not occurring. As hyperparameters are properly set.

Results and Discussion

In this section, the results of different models are given and discussed on four performance parameters: accuracy, precision, recall, and F1-score. Manual features showed

Table 6 Performance of DL on news headlines

Input text feature	Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Handcrafted feature	CNN	85.00	85.43	75.93	80.40
	LSTM	74.05	74.00	72.80	73.40
	BiLSTM	76.51	75.43	71.00	73.15
	CNN + LSTM	77.00	76.90	74.56	75.71
	CNN + BiLSTM	79.56	77.00	78.73	77.86
Word embedding feature	CNN	84.14	85.67	78.18	81.75
	LSTM	83.37	87.42	75.85	81.23
	BiLSTM	83.57	84.4	80.17	82.23
	CNN + LSTM	84.19	83.28	83.42	83.35
	CNN + BiLSTM	85.32	84.56	83.42	83.99

Table 7 Performance of DL on Reddit

Input text feature	Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Handcrafted feature	CNN	87.90	87.32	86.09	86.70
	LSTM	78.90	77.56	78.21	77.88
	BiLSTM	79.03	78.67	77.54	78.10
	CNN + LSTM	80.06	79.03	77.89	78.46
	CNN + BiLSTM	81.23	78.90	79.45	79.17
Word embedding feature	CNN	91.23	87.88	89.67	88.77
	LSTM	91.32	87.90	85.34	86.60
	BiLSTM	90.72	88.02	86.89	87.45
	CNN + LSTM	92.00	90.01	89.87	89.94
	CNN + BiLSTM	92.01	89.56	91.01	90.28

Table 8 Performance of DL on Twitter

Input text feature	Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Handcrafted feature	CNN	80.02	81.04	78.89	79.95
	LSTM	81.22	80.76	77.93	79.32
	BiLSTM	81.00	81.07	78.23	79.62
	CNN + LSTM	81.90	81.24	79.8	80.51
	CNN + BiLSTM	81.96	78.04	79.05	78.54
Word embedding feature	CNN	87.34	87.20	87.90	87.75
	LSTM	85.71	85.60	86.29	85.94
	BiLSTM	85.30	84.92	86.29	85.60
	CNN + LSTM	86.32	87.55	85.05	86.29
	CNN + BiLSTM	92.01	89.56	91.01	90.28

Table 9 Results of DL models on News Headlines considering each feature

	Models Used				
Features used	CNN	LSTM	BiLSTM	CNN+ LSTM	CNN+BiLSTM
POS	69.96	70.05	69.08	68.09	68.75
POS + stopword	72.39	72.50	73.00	71.02	72.89
Negation	52.79	55.05	60.30	56.78	58.06
POS + negation	71.16	72.90	73.29	70.10	71.25
N-gram	80.52	75.90	81.02	80.20	81.02
POS + negation + N-gram	83.33	79.56	84.11	78.45	79.56

significant results for both machine learning and deep learning classifiers. Table 5 shows the results of machine learning classifiers. The Random Forest classifier outperformed other classifiers for all the datasets. To verify how deep learning models performed, a common strategy was used: the initial model was trained with a different number of records, accuracy was checked, and the setup that gave the best accuracy was finalized.

Tables 6, 7, 8 show the performance of deep neural architecture on three datasets. The CNN with handcrafted features over performed on news headlines and Reddit dataset, whereas hybrid model worked better on twitter dataset. If compared with word embedding feature vector, CNN and hybrid model worked well for all the 3 datasets.

To further see how every handcrafted feature contributes to performance, a study was done for classifying text of news headlines. As some of the features are one hot encoding, it did not perform well. Table 9 shows the accuracy of features used. It is very much evident that N-gram feature worked very well on all the models. Reason might be as it has real values and all others are integer values. However, when used all features except stopword the results are near to the model considering all the features. To improvise the models performance, some more experimentations were done using pretrained word embedding technique to find how much variation can be seen in the results generated. For this Google GloVe pretrained embedding is used. As word embedding just finds the similarity between the words used in the sentences and gives the score accordingly, it does not take into consideration the context of the words in the sentence. As words used in the sentence can mean differently based on the adjacent words. Hence, context of the sentence needs to be understood. Therefore, to verify how context details can help in classification task, BERT model with base setting is used. It is further fine-tuned for the classification task. Till now, the experiments were done using words in isolation or making use of patterns. Now, Table 10 shows result of deep learning models with pretrained word embedding and language model BERT. If compared with simple word embedding, GloVe has outperformed and similarly BERT has worked very well on News headlines dataset and Reddit, whereas on Twitter, it has outperformed. It is very much clear that twitter text requires more contextual understanding for classification task as it is not written by experts, whereas text in news headline and reddit dataset is written by experts.

Overall study shows that manual features used are useful and significant in classifying sarcasm as compared as to other approaches used. Table 11 shows comparison of proposed model with the existing work. It is evident that our model performed marginally similar to other models.

Table 10 Results of GloVe and BERT model on 3 datasets

Dataset	Models	CNN	LSTM	BiLSTM	CNN+ LSTM	CNN+ BiLSTM	BERT
News headlines	Accuracy	78.38	80.62	85.09	85.48	85.74	82.00
	Precision	78.32	80.30	83.75	84.34	86.59	80.88
	Recall	76.94	79.45	86.12	84.04	80.63	82.45
Reddit	Accuracy	91.02	91.67	91.34	92.34	92.55	91.72
	Precision	89.76	85.69	86.89	90.25	91.56	87.32
	Recall	85.87	88.34	89.45	91.34	92.00	95.69
Twitter	Accuracy	86.21	75.02	91.47	91.50	80.22	91.22
	Precision	90.05	73.89	93.43	92.67	93.59	95.69
	Recall	82.69	77.02	89.42	81.56	84.13	87.80

Table 11 Performance comparison of proposed with existing work

Author	Method	Dataset	Accuracy (%)	Precision (%)	Recall (%)	F1score (%)
Bhakuni et al. [26]	SVM	Twitter	93.00	–	–	–
Jain et al. [27]	BiLSTM + CNN	Twitter	92.71	–	–	89.05
Kumar et al. [28]	BiLSTM+ Attention	Twitter	82.49	83.25	83.13	83.18
		Reddit	84.16	84.31	83.25	84.31
Sharma et al. [29]	AutoEncoder	Headline	90.81	92.00	91.02	91.00
		Twitter	92.80	95.00	86.01	90.00
Proposed Model 1 (using handcrafted feature input)	CNN	Headline	85.00	85.43	75.93	80.40
	CNN	Reddit	87.90	87.32	86.09	86.70
	CNN+ BiLSTM	Twitter	81.96	78.04	79.05	78.54
Proposed Model 2 (using word embedding feature input)	CNN+BiLSTM	Headline	85.32	84.56	83.42	83.99
		Reddit	92.01	89.56	91.01	90.26
		Twitter	87.34	87.20	87.90	87.75

Conclusion and Future Directions

Sarcasm detection is a sub-branch of sentiment analysis. Sarcasm can change the polarity of the sentiment. Hence, it is detected than proper understanding of the text can be interpreted; which in turn will help people to understand the text correctly. Sarcasm in news or in product review may mislead the user expression or reaction about it. Sarcastic news, tweets, posts, and narratives are prevalent in today's social media era. Sarcasm detection and sentiment analysis are closely related tasks; hence, sarcasm detection is an important issue to ponder upon for both industry professionals and academicians. Research can be done more on understanding the context of the text which can easily help to find the way sarcastic news are written. Hybrid model based on content and context can be employed which will help researchers to correctly identify sarcasm. As less data are annotated, so in future can explore unsupervised or semi-supervised models to find the labels and than perform classification.

Author Contributions All authors have contributed equally to this work.

Funding Not applicable.

Data Availability Statement Datasets are available publicly. The links for the datasets are 1. https://www.kaggle.com/datasets/danofer/sarcasm, 2. https://www.kaggle.com/rmisra/news-headlines-dataset-for-sarcasm-detection/kernels, and 3. https://www.kaggle.com/c/gse00.

Declarations

Conflict of Interest The authors declare that they have no conflict of interest.

Ethics Approval This is to acknowledge that this manuscript is original, has not been published before, and is not currently being considered for publication elsewhere.

References

1. Stringfellow FJ. Meaning of irony. The: a psychoanalytic investigation. New York: State University of New York Press; 1994.
2. Colston H, Gibbs R. A brief history of irony, irony in language and thought: a cognitive science reader. 2007. pp. 3–21.
3. Kreuz RJ, Glucksberg S. How to be sarcastic: the echoic reminder theory of verbal irony. J Exp Psychol Gen. 1989;118(4):374–86.
4. Campbell J, Katz AN. Are there necessary conditions for inducing a sense of sarcastic irony? Discourse Process. 2012;49(6):459–80.
5. Kumar Y, Goel N. Ai-based learning techniques for sarcasm detection of social media tweets: state-of-the-art survey. SN Comput Sci. 2020;1(6):318–31.
6. Amir S, Wallace B, Lyu H, Carvalho P, Silva M. Modeling context with user embeddings for sarcasm detection in social media. In: Proceedings of the 20th SIGNLL conference on computational natural language learning, 2016. pp. 167–77.
7. Zhang M, Zhang Y, Fu G. Tweet sarcasm detection using deep neural network. In: Proceedings of COLING, the 26th international conference on computational linguistics: technical papers, 2016. pp. 2449–60.
8. Ghosh A, Veale T. Magnets for sarcasm: making sarcasm detection timely, contextual and very personal. In: Proceedings of the 2017 conference on empirical methods in natural language processing, 2017. pp. 482–91.

9. Cambria E, Poria S, Hazarika D, Kwok S. Discovering conceptual primitives for sentiment analysis by means of context embeddings. In: Proceedings of the thirty-second AAAI conference on artificial intelligence, (AAAI-18), the 30th innovative applications of artificial intelligence (IAAI-18), and the 8th AAAI symposium on educational advances in artificial intelligence (EAAI-18), 2018. pp. 1795–802.
10. Pelser D, Murrell H. Deep and dense sarcasm detection (2019). arXiv:1911.07474.
11. Kumar A, Narapareddy VT, Srikanth VA, Malapati A, Neti LBM. Sarcasm detection using multi-head attention based bidirectional lstm. IEEE Access. 2020;8:6388–97.
12. Razali MS, Halin AA, Ye L, Doraisamy S, Norowi NM. Sarcasm detection using deep learning with contextual features. IEEE Access. 2021;9:68609–18.
13. Bharti SK, Gupta RK, Shukla PK, Hatamleh WA, Tarazi H, Nuagah SJ. Multimodal sarcasm detection: a deep learning approach. Wirel Commun Mob Comput. 2022;2022.
14. Baroiu A, Matu S. Comparison of deep learning models for automatic detection of sarcasm context on the mustard dataset. Electronics. 2023;12(3).
15. Davidov D, Tsur O, Rappoport A. Semi-supervised recognition of sarcasm in twitter and amazon. In: Proceedings of the fourteenth conference on computational natural language learning, 2010. pp. 107–16.
16. Nozza D, Fersini E, Messina E. Unsupervised irony detection: a probabilistic model with word embeddings. 2016. pp. 68–76.
17. Mukherjee S, Bala PK. Sarcasm detection in microblogs using naïve bayes and fuzzy clustering. Technol Soc. 2017;48:19–27.
18. Potamias RA, Siolas G, Stafylopatis AG. A transformer-based approach to irony and sarcasm detection. Neural Comput Appl. 2020;32:17309–20.
19. Oprea S, Magdy W. Exploring author context for detecting intended vs perceived sarcasm. In: Proceedings of the 57th annual meeting of the association for computational linguistics, 2019. pp. 2854–9.
20. Rajadesingan A, Zafarani R, Liu H. Sarcasm detection on twitter: a behavioral modeling approach. In: Proceedings of the eighth ACM international conference on web search and data mining, 2015. pp. 97–106.
21. Kadyan V, Bawa P, Hasija T. In domain training data augmentation on noise robust punjabi children speech recognition. J Ambient Intell Humaniz Comput. 2022:2705–21.
22. Shetty V, Singh M, Salunkhe S, Rathod N. Comparative analysis of different classification techniques. SN Comput Sci. 2022;3(1).
23. Yassin M, Mahmuddin M. The effect of pre-processing techniques on the accuracy of sentiment analysis using bag-of-concepts text representation. SN Comput Sci. 2021;2(4).
24. Kaur H, Kadyan V. Feature space discriminatively trained punjabi children speech recognition system using kaldi toolkit. In: Proceedings of the international conference on innovative computing & communications, 2020.
25. Khodak M, Saunshi N, Vodrahalli K. A large self-annotated corpus for sarcasm. In: Proceedings of the eleventh international conference on language resources and evaluation-LREC, 2018.
26. Bhakuni M, Kumar SK, Garg, Iwendi C, Singh A. Evolution and evaluation: Sarcasm analysis for twitter data using sentiment analysis. J Sens. 2022;2022.
27. Jain D, Kumar A, Garg G. Sarcasm detection in mash-up language using soft-attention based bi-directional lstm and feature-rich cnn. Appl Soft Comput. 2020;91.
28. Kumar P, Sarin G. Welmsd-word embedding and language model based sarcasm detection. Online Inf Rev. 2022;46(7):1242–56.
29. Sharma D, Singh B, Agarwal S, Pachauri N, Alhussan A, Abdallah H. Sarcasm detection over social media platforms using hybrid ensemble model with fuzzy logic. Electronics. 2023;12(4).