

Sarcasm Detection using Contextual Embeddings

A Project Report
Submitted for the partial fulfilment for the award of the degree of

**Bachelor of Technology
in
Computer Science / Information Technology**

Submitted by:

**[GroupID: CSIT06]
Aahna Sethi [2116091]
Akanksha Sharma [2116728]
Ananya Singh Gautam [2116104]
Sakshi [2116814]**

Under the supervision of

Dr. Nisheeth Joshi
Associate Professor
Department of Computer Science
Banasthali Vidyapith
Rajasthan.



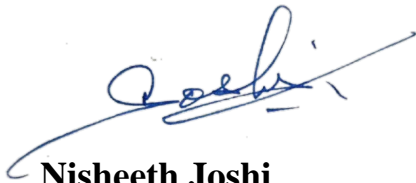
Department of Computer Science

Banasthali Vidyapith

Session: 2023-24

Certificate

Certified that **Aahna Sethi, Akanksha Sharma, Ananya Singh Gautam and Sakshi** has carried out the project work titled “**Sarcasm Detection using Contextual Embeddings**” from **August ‘23** to **April ‘24** for the award of the **Bachelor of Technology** from **Banasthali Vidyapith** under my supervision. The thesis embodies result of original work and studies carried out by Student herself and the contents of the thesis do not form the basis for the award of any other degree to the candidate or to anybody else.



Nisheeth Joshi

Designation: Associate Professor, Department of Computer Science

Place: Banasthali Vidyapith, Rajasthan

Date: 3rd April, 24

ABSTRACT

This study delves into the intricate challenge of sarcasm detection, given its subtle and context-dependent nature in natural language processing. Our investigation focuses on the efficacy of employing contextual embeddings to discern the nuanced semantics embedded within sarcastic statements. Through the integration of contextual embeddings with LSTM networks, we harness the power of sequential modelling and contextual richness to interpret sarcasm within diverse linguistic contexts. By conducting empirical analyses and experiments, we explore the mutual relationship between contextual embeddings and LSTM networks, illuminating the complex mechanisms underpinning contextual understanding and sequential reasoning in textual data analysis.

Our research not only achieves great performance in sarcasm detection across news headlines dataset but also yields invaluable insights for the broader domains of natural language processing and sarcasm detection. We showcase the potential of contextual embeddings to increase sarcasm identification, thereby presenting implications for various applications in sentiment analysis and natural language understanding.

Keywords: Natural Language Processing, LSTM, Contextual Embeddings, Sarcasm Detection

ACKNOWLEDGEMENT

First and foremost, we would like to express our deep sense of gratitude to our mentor, Dr. Nisheeth Joshi, Associate Professor, Department of Computer Science, Bansthali Vidyapith, Rajasthan for his constant guidance and direction which facilitate us to complete the project. We are indebted for his constant help and encouragement throughout the preparation. We are sincerely thankful for the invaluable support by our senior, Dr. Pragya Katyayan, PhD (CS), Senior Research Fellow (SERB), Department of Computer Science, Banasthali Vidyapith, Rajasthan.

We are highly obliged to sincerely thank Prof. Chandra Kumar Jha (Professor and Head, Department of Computer Science), Dr. Neelam Sharma (Associate Professor) and Dr. Deepak Kumar (Associate Professor) for the help they rendered whenever needed. We would also like to extend our gratitude to the Department of Computer Science (Apaji Institute) and Centre for Artificial Intelligence, Banasthali Vidyapith for providing the resources whenever needed.

Finally, we would like to thank our families and friends for their continuous encouragement and support throughout this journey.

Aahna Sethi

Akanksha Sharma

Ananya Singh Gautam

Sakshi

TABLE OF CONTENT

1. INTRODUCTION	7
2. LITERATURE REVIEW	21
3. METHODOLOGY	21
3.1 Corpus Creation	21
3.2 Extraction of Linguistic and Cognitive Features	22
3.3 Deep Learning Model	22
3.4 LSTM Model	24
3.5 Workflow	26
4. RESULT AND DISCUSSION	28
4.1 Quantitative Analysis	28
4.2 Qualitative Analysis	32
5. CONCLUSION	33
6. APPENDICS	39
7. REFERENCES	44
8. LIST OF PUBLICATION	48

INTRODUCTION

In recent years, research on the detection of sarcasm through natural language processing (NLP) and deep learning techniques has gained significant importance. Because sarcasm is nuanced and frequently context-dependent, it poses a special challenge in computational linguistics. Sarcasm is defined as the use of irony to convey contempt or ridicule. We explore approaches and developments in sarcasm detection in this introduction, with special attention to the use of contextual embeddings and Long Short-Term Memory (LSTM) networks.

Sarcasm, a type of verbal irony in which the intended meaning frequently contradicts the literal interpretation of the words used, has long been acknowledged as a difficulty in comprehending natural language. There is a great deal of interest in creating computational techniques to recognize and comprehend sarcastic expressions because sarcasm is widely used in social media, online forums, and casual conversations. Because sarcasm is subtle, ambiguous, and dependent on context, it can be challenging to identify in text. In contrast to direct statements, sarcastic remarks frequently necessitate comprehension of the speaker's tone, intention, and larger context. For instance, depending on the situation and the speaker's tone, the phrase "Oh, great job!" could be taken as sarcastic criticism or sincere praise.

The cornerstones of conventional sarcasm detection methods have been shallow linguistic traits and rule-based systems. Sadly, these methods typically struggle to fully capture the complexity and context-dependency of sarcastic expressions, which restricts their accuracy and generalizability. As a result, there's been a lot of interest in applying advanced natural language processing (NLP) methods and deep learning models to improve sarcasm detection.

Sarcasm is inherently ambiguous and variable because it frequently relies on minute language cues like tone, intonation, and context. For example, depending on the speaker's tone and the context in which it is used, the statement "Oh, great job!" could be taken as sarcastic criticism or sincere praise. Computational models have a great deal of difficulty in accurately interpreting the intended meaning of sarcastic statements because of this ambiguity.

Sarcasm also depends a great deal on the context in which it is used, such as the speaker's identity, the interpersonal dynamics between interlocutors, and the larger social and cultural background. Accurate outcomes in a variety of communication contexts depend on capturing and integrating this contextual data into sarcasm detection algorithms. Linguistically, sarcastic statements can be expressed in a variety of ways, such as through lexical ambiguity, irony, understatement, or hyperbole. Furthermore, in order to express its intended meaning, sarcasm frequently employs negation, double entendre, and other rhetorical strategies. Computational models face an additional challenge due to linguistic complexity, as they need to accurately identify and interpret these nuanced linguistic cues.

Sarcasm is also intrinsically subjective, necessitating an awareness of the speaker's motivations, attitudes, and feelings. Certain individuals may interpret something as sarcastic in a different way than others due to differences in background, beliefs, and cultural norms. Robust sarcasm detection thus requires models that can take into account the subjective nature of sarcasm and adjust to various speaker intentions. Furthermore, class imbalance and size constraints are common in annotated datasets for sarcasm detection, where sarcastic examples are less common than non-sarcastic ones.

Due to the lack of labelled data, deep learning architectures—which normally need a lot of labelled data to function at their best—find it difficult to train sarcasm detection models that are both accurate and broadly applicable. The conventional methods for detecting sarcasm have predominantly depended on rule-based frameworks and superficial linguistic characteristics. These techniques frequently make use of custom rules and heuristics to pinpoint language patterns frequently connected to sarcasm. To detect the presence of sarcasm, for instance, certain rule-based systems may search for particular linguistic markers like negation, exaggeration, or incongruity. In conventional sarcasm detection techniques, shallow linguistic features like word frequency, part-of-speech tags, and syntactic patterns have also been used. To distinguish between sarcastic and non-sarcastic statements, these textual characteristics are taken out and fed into machine learning classifiers, like decision trees or support vector machines (SVM).

Nevertheless, the intricacy and context-dependency of sarcastic expressions are frequently beyond the scope of conventional sarcasm detection techniques. Because rule-based systems rely on predefined rules that might not translate well to various contexts or linguistic variations, they may have limitations. The finer points of sarcasm can also be lost on shallow language features, especially when the sarcasm heavily depends on speaker intention or contextual clues.

Moreover, conventional methods are less scalable and flexible to new datasets or domains since they frequently involve manual feature engineering and are labour-intensive. Consequently, these techniques might have poor generalizability and accuracy, particularly when used on diverse and dynamic language data sets like social media or online forums.

The use of deep learning models and sophisticated natural language processing (NLP) techniques for sarcasm detection has increased in the past few years. These methods, by capturing the contextual nuances of sarcasm and learning representations directly from data, offer the potential to overcome many of the drawbacks of traditional approaches, which we will discuss in the following sections.

The field of sarcasm detection has changed dramatically as a result of advances in natural language processing (NLP), opening up new avenues for more complex and precise approaches to handling this challenging linguistic phenomenon. The creation of contextual embeddings, such as BERT (Bidirectional Encoder Representations from Transformers), ELMo (Embeddings from Language Models), and GPT (Generative Pre-trained Transformer), has been one of the major advances in recent years. By capturing the meaning of words and phrases in context, these contextual embeddings enable models to comprehend language's finer points, such as sarcasm.

Because BERT can capture bidirectional context and generate rich contextual representations of text, it has been widely used in sarcasm detection research. BERT-based models are capable of efficiently utilizing contextual information to recognize sarcastic expressions in a variety of linguistic contexts by pre-training on a large corpora of text data and fine-tuning on sarcasm detection tasks. Similarly, deep contextualized representations learned from bidirectional language models are

incorporated into ELMo embeddings, which have demonstrated promise in capturing the contextual subtleties of sarcasm and enhancing detection accuracy.

An important development in natural language processing (NLP) for sarcasm detection is the application of deep learning architectures, like transformers and Long Short-Term Memory (LSTM) networks. Because LSTM networks can capture long-range dependencies and model sequential data, they are a good fit for tasks that require contextual understanding, like sarcasm detection. Researchers have achieved state-of-the-art performance in sarcasm detection tasks by combining the strengths of both approaches: contextual embeddings and LSTM architectures.

Furthermore, by enabling large-scale pre-training of language models on massive amounts of text data, transformers—like the GPT series—have revolutionized natural language processing (NLP). To improve performance, these pre-trained transformer models can be refined for particular tasks, like sarcasm detection, by utilizing their acquired representations. Transformers are incredibly useful for sarcasm detection in challenging linguistic contexts because of their self-attention mechanism, which enables them to effectively capture long-range dependencies and contextual information.

Overall, sarcasm detection systems can now detect subtle nuances in sarcastic language and achieve higher levels of accuracy and robustness thanks to advancements in natural language processing (NLP). The state-of-the-art in sarcasm detection has advanced significantly as a result of researchers' use of contextual embeddings, deep learning architectures, and semantic parsing techniques to overcome obstacles.

Neural networks have made significant progress with the development of Long Short-Term Memory (LSTM) networks, especially in the area of natural language processing (NLP). Long-term dependency learning (LSTM) is a unique feature of Recurrent Neural Networks (RNNs), which Hochreiter and Schmidhuber developed in 1997. This ability solves the vanishing gradient problem, which prevents traditional RNNs from effectively learning from lengthy input sequences. New opportunities for processing and comprehending sequential data, like text, where context and order are essential, have been made possible by the advent of LSTMs into the field of natural language processing (NLP).

In LSTM gates control the flow of information in and out of the cell state. As the sequence goes on, these gates—the forget gate, input gate, and output gate—decide which information is crucial to retain or discard. The input gate chooses what new data to add, the forget gate chooses what portion of the current cell state to discard, and the output gate chooses which portion of the current cell state to send to the following layer or step. Because of this complex mechanism, LSTMs are very good at complex sequential tasks because they can decide which information to remember or forget.

The capacity of LSTMs to comprehend context and retain details from previous passages in the text is extremely useful for sarcasm detection. Since sarcasm frequently depends on inferential clues and prior knowledge, it is necessary for a model to interpret not only words or phrases but also their meanings in relation to the sentence or conversation as a whole. For example, determining a text's sarcastic tone may require comparing a sentence's sentiment to the context in which it appears overall—a task that LSTM networks are well-suited to. Long Short-Term Memory networks are pivotal in machine learning and NLP, enabling efficient data processing and sarcasm detection, demonstrating their ability to understand context-dependent human language.

The growing field of sarcasm detection demonstrates the broad implications of identifying sarcasm in text, as it finds application not only in the theoretical domains of natural language processing (NLP) but also in a variety of real-world scenarios. For example, in social media analytics, sarcasm detection accuracy is critical to comprehending public opinion about products, services, and brands. This nuanced understanding helps businesses make better decisions based on the underlying sentiments expressed in user-generated content, enhance customer service, and refine their marketing strategies.

Similar benefits accrue to the field of human-computer interaction (HCI) from advances in sarcasm detection. The ability of conversational agents and chatbots to detect and react appropriately to sarcasm is essential for promoting natural and productive interactions as these tools become more and more integrated into our daily lives. This feature guarantees that these systems are able to have more authentic conversations that are sensitive to the user's intent, improving the user experience as a whole.

Furthermore, the field of content moderation and cyberbullying detection benefits greatly from the ability to detect sarcasm. Satire is often used as a cover for hate speech and other negative behaviors, which pose a challenge to online communities and platforms. Content moderation efforts can be greatly aided by automated tools that can recognize sarcasm. These tools can help identify harmful content and make online spaces safer and more inclusive.

Sarcasm detection has applications in literary analysis and cultural studies as well. In these fields, it is a useful tool for analyzing texts and deciphering subtleties in communication between various authors, genres, and cultural contexts. This analysis enhances the study of literature and culture by providing insights into stylistic devices and communication patterns. Ultimately, sarcasm detection is important in multilingual and cross-cultural communication because it is a challenge that transcends linguistic and cultural boundaries. The need for technologies that can comprehend and adjust to the nuances of sarcasm in various languages and cultural contexts is growing more pressing as globalization picks up speed. Global businesses, translators, and international communication platforms that want to enable precise and transparent interactions worldwide need these kinds of capabilities.

When taken as a whole, these applications highlight how important sarcasm detection is to improving digital communication, encouraging safer online spaces, and overcoming linguistic and cultural barriers. The potential for sarcasm detection to benefit a wide range of industries—from business and technology to culture and social interaction—remains enormous as this field develops, suggesting a future in which comprehending the complexities of human language will enhance both our virtual and offline experiences.

LITERATURE REVIEW

Riloff et al. (2013) developed Sarcasm as Contrast between a Positive Sentiment and Negative Situation. They worked on the contrast between the situation and the sentiment of the author. For identifying contrast between emotion and sarcasm they developed a sarcasm recognizer. They presented an algorithm that accordingly learns positive sentiment phrases and negative situation phrases from sarcastic tweets. And showed that the algorithm yielded better recall for sarcasm recognition. Their study had limitations like the phrase they used was limited to the same type structure and context. And they wanted to learn more on how to identify stereotypical negative activity because it is essential to have that world knowledge to identify sarcasm. If we can identify the intensity of the negative situation then also it is useful to differentiate between positive and negative contrast

Maynard et al. (2013) presented an integrated method for social media analysis that integrates textual and multimedia data. The study suggested a rule-based textual method enhanced by multimedia analysis in light of the difficulties brought on by the dynamic nature of social web material, including noisy text and complicated multimedia interactions. The research emphasised the significance of structured preservation based on semantic categories along with emphasis on helping archivists choose pertinent content for communal memory preservation. The article, which offers promising insights into fields including sentiment analysis, sarcasm detection, and discourse analysis, is notable for introducing unique text and multimedia mining algorithms customised to the nuances of social media. The modular design of the study paved the way for future developments in comprehending and examining the complex world of social media interactions.

Ptacek et al. (2014) conducted a study on sarcasm detection in both English and Czech tweets using a machine learning approach. They assessed two classifiers with diverse feature combinations on datasets from both languages. The study addressed the complexities of Czech morphology by exploring various preprocessing techniques. Notably, their results demonstrated the superiority of a language-independent approach over an adapted state-of-the-art method in English (F-measure 0.947). The research primarily emphasised the rich morphology and syntax of both languages, especially

Czech, and incorporated language-independent methods and preprocessing recommendations from Habernal et al. (2013). Overall, their work focused on document-level sarcasm detection through supervised machine learning and contributed by extensively evaluating classifiers, feature sets, and preprocessing techniques for both Czech and English datasets.

Nagwanshi and Madhavan (2014) presented an innovative approach for detecting sarcasm in text, focusing on key features such as semantic content and sentiment reversal. They supplemented their findings with human evaluation results and aimed to enhance their approach by integrating both linguistic and cognitive aspects of text processing. Recognizing the complexity of automatically processing sarcastic statements, they introduced a novel technique that relies on semantics and the length of sentiment progressions. Their research demonstrated the feasibility of distinguishing between sarcastic and negatively toned statements with an accuracy of up to 75%. Although the set of features proposed may not represent the optimal differentiator, the framework they developed holds promise for incorporating additional semantic and cognitive elements into sarcasm detection.

Liu et al. (2014) delved into the nuances of sarcasm within both English and Chinese languages. Detecting sarcasm holds substantial significance for various Natural Language Processing (NLP) applications, such as sentiment analysis, opinion mining, and advertising. They introduced an innovative ensemble learning approach known as the Multi-strategy Ensemble Language Approach (MSELA), to address the challenge of class imbalance. They evaluated their model on datasets in both English and Chinese, their experiments demonstrated superior performance compared to existing sarcasm detection methods and imbalanced classification techniques. Sarcasm, classically defined as the deliberate use of words to convey a meaning contrary to their literal interpretation, served as their focal point. They further introduced an explicit sarcasm feature set, aiming to capture distinctive characteristics of sarcasm in both languages, encompassing low-level and high-level properties. Through a series of experiments, they compared their model with existing approaches, with the ultimate goal of automating sarcasm annotation and uncovering new features, enabling the creation of an enhanced model capable of detecting sarcasm across a variety of textual contexts.

Rajadesingan et al. (2015) developed a novel approach to detect sarcasm on Twitter by incorporating the behavioural traits of users expressing sarcasm. It utilised users' past tweets and drew from behavioural and psychological theories to construct a behavioural modelling framework. Unlike prior studies focusing solely on linguistic cues, this approach combined content analysis with user behaviour analysis. The paper formulated the sarcasm detection problem as discerning whether an unlabeled tweet from a user, accompanied by that user's past tweets, is sarcastic. The proposed SCUBA framework (Sarcasm Classification Using Behavioral Modeling Approach) evaluates the likelihood of a user being sarcastic and employs supervised learning algorithms with constructed features to detect sarcastic tweets. Through experiments, the authors demonstrate SCUBA's effectiveness in identifying sarcastic tweets. SCUBA stood out for considering psychological and behavioural aspects of sarcasm and leveraging historical information, even with limited data, enhancing sarcasm detection efficiency. These findings made SCUBA suitable for real-time applications with computational constraints.

Kunneman et al. (2015) discovered that the inclusion of hashtags effectively reduced the necessity for employing other linguistic indicators such as exclamations and intensifiers to convey sarcasm. They proposed that these explicit markers, like hashtags, function as digital counterparts to non-verbal expressions commonly used in face-to-face interactions to convey sarcasm. Their research involved the development and testing of a system designed to identify sarcastic tweets within a sample of Dutch tweets posted on a single day. Their findings indicated that distinguishing between sarcastic and literally intended tweets proved to be quite challenging. They observed that most tweets carried a genuinely positive message and identified four categories of markers associated with sarcasm: intensified and non-intensified evaluative words, exclamations, and non-sarcastic hashtags. It was noted that intensified evaluative words and exclamations induced hyperbole, yet they occurred less frequently in sarcastic tweets compared to non-intensified evaluative words. While their study shed light on the utilisation of sarcasm in Twitter messages and successfully trained a machine learning classifier for automated detection, they emphasised their focus on hyperbole as a primary linguistic marker used for sarcasm. Notably, they found that typical hyperbole inducers, including exclamations and intensifiers, ranked among the most predictive features of their classifier.

Bamman and Smith (2015) emphasised sarcasm's dependence on shared knowledge between speakers and audiences, highlighting its contextual nature. They demonstrated that incorporating non-linguistic context data from Twitter, including author and audience properties, improved sarcasm detection accuracy over linguistic features alone. Sarcasm's effectiveness hinged on the principle of inferability, as per Kreuz (1996), where speakers used sarcasm when confident it will be understood, typically more among acquaintances than strangers. The author-audience relationship is pivotal, but it becomes intricate on social media, where the audience is unknown or vague. Users often add #sarcasm when unfamiliar with their audience, signalling intent to those who may not discern sarcasm otherwise. Their model used Tweet, Author, Audience, and Response features, where Author features provided the greatest accuracy improvement, reaffirming the role of author-audience interaction in sarcasm recognition. The #sarcasm hashtag served as a communication tool rather than a natural indicator and aided the audience in interpreting sarcasm. Detecting sarcasm without explicit markers required alternative approaches.

Zhang et al. (2016) explored the application of a neural network for detecting sarcasm in tweets. They conducted a comparison between continuous automatic features and discrete manual features in their analysis. Their approach involved employing a bi-directional Gated Recurrent Neural Network (GRNN) to capture both syntactic and semantic information in tweets, along with a pooling neural network to automatically extract contextual features from past tweets. To assess the performance of the neural network, they compared it with a traditional discrete model. They first created a baseline discrete model, incorporating commonly used features from previous literature, such as characteristics of the target tweet content and historical tweets from the same author. They used a GRNN to model tweet content and employed a gated pooling function for feature extraction. To represent significant words from contextual tweets, they utilised pooling for direct feature extraction. Their deep neural network model for tweet sarcasm detection was innovative in that it eliminated the need for manual feature engineering and external resources like POS taggers and sentiment lexicons. Instead, it leveraged distributed embedding inputs and recurrent neural networks to induce semantic features. The neural network model outperformed the state-of-the-art discrete model, demonstrating improved results.

Bouazizi and Ohtsuki (2016) developed a pattern-based approach for sarcasm detection on Twitter. Sarcasm is an ironic form of criticism or teasing, which is even difficult for humans to decipher. They used the idea of sentiment analysis to group the opinions of internet users on a specific topic. They highlighted the necessity of pattern-based features to distribute tweets as sarcastic and non-sarcastic. Sarcasm detection's necessity and effectiveness depend on accurately identifying sarcasm in sarcastic statements, as per Maynard and Greenwood (2014). Their approach took into account the different types of sarcasm and analysed the tweets regardless of their temporal context. They uncovered the key purposes of sarcasm in social networks and proposed an efficient method to detect tweets. They utilised part-of-speech tags to extract patterns that define the degree of sarcasm in tweets. Their goal was to enhance sentiment analysis accuracy by using this information.

Amir et al. (2016) introduced a novel deep neural network that effectively recognizes and leverages user embeddings in conjunction with lexical cues to identify sarcasm. Their method did not rely on intricate feature engineering; rather, it solely utilised data from users' prior posts. They observed that earlier computational techniques often relied on surface-level patterns in the frequency of specific words, whereas they recognized that the interpretation of a given statement as literal or sarcastic depends heavily on the speaker. To address this, they proposed a fresh approach to sarcasm detection that eliminated the need for extensive manual feature engineering. Their method centred around a neural model that harnessed both content embeddings and context embeddings. They incorporated a vector representation of lexical cues using a convolutional layer. Subsequently, their model acquired user embeddings. This combined technique resulted in the creation of CUE-CNN; a deep neural network designed to broadly analyse sarcastic comments on social media.

Khodak et al. (2017) introduced the Self-Annotated Reddit Corpus (SARC), a groundbreaking dataset for sarcasm research housing 1.3 million self-labelled sarcastic statements annotated with "/s". Diverging from previous datasets, SARC offers a rich tapestry of data, embedding both self-annotations and contextual intricacies like user profiles, topics, and conversation context. Encompassing a staggering 533 million comments, this corpus stands as a beacon for large-scale machine learning endeavours. Confronting challenges such as noise stemming from the "/s" marker, the authors

meticulously validate SARC's credibility by comparing it with sources like Twitter and the Internet Argument Corpus. Through meticulous manual evaluations, subtle false positive and false negative nuances emerge. While baseline methods like Bag-of-n-Grams and Sentence Embeddings strive, human evaluators shine, underscoring the complexity of the sarcasm detection task. This paper, heralding a new era, not only presents a substantial resource but also paves the way for the development of cutting-edge sarcasm detection algorithms.

Joshi et al. (2017) presented "Automatic Sarcasm Detection: A Survey" presents a comprehensive overview of computational approaches to detect sarcasm in text. It highlights the complexities of this problem due to the varied ways sarcasm is expressed. The paper is significant for being the first compilation of previous research in this field, identifying three key milestones in sarcasm detection: semi-supervised pattern extraction, hashtag-based supervision, and the incorporation of context beyond the target text. The paper also categorises the approaches used in sarcasm detection, such as rule-based methods that rely on predefined rules, and statistical approaches that use features like sentiment changes. It emphasises the importance of understanding the relationship between sentiment and sarcasm and addresses the issue of data skew in sarcasm-labelled datasets.

Chaudhari and Chandankhede (2017) explored diverse methodologies, ranging from rule-based techniques leveraging hashtags and specific indicators, to statistical approaches integrating lexical, pragmatic, and pattern-based features. Distributional methods, rooted in the Distributional Hypothesis, employ semantic contexts for nuanced understanding. Machine learning classifiers like SVM and deep learning techniques, especially multimodal fusion of textual and visual data, have emerged as powerful tools. Rule-based methods offer precision, statistical approaches blend features for accuracy, and distributional methods provide semantic depth. Machine learning classifiers discern patterns effectively, while deep learning, through multimodal fusion, captures intricate nuances. As research advances, the integration of diverse datasets, innovative feature sets, and hybrid models combining rule-based, statistical, and deep learning approaches are poised to shape the future of sarcasm detection, expanding its applications in sentiment analysis and social media analysis

Bharti et al. (2017) developed Sarcasm Analysis on Twitter Data Using Machine Learning Approaches. Sarcasm detection is one of the challenging tasks in the field of NLP. They proposed four approaches namely parsing-based lexical generation algorithm, likes and dislikes contradiction, tweet contradicting universal facts, and tweet contradicting temporary facts. After that they also deployed four machine learning classifiers namely support vector machine, Naive Bayes, maximum entropy, and decision tree. Classifiers were trained using extracted features Their work achieved considerable accuracy. Highest accuracy was achieved by PBLGA approach Naive Bayes performed worst out of all the other remaining approaches.

Misra and Arora (2018) examined sarcasm recognition in natural language processing and highlighted the shortcomings of earlier research that used Twitter-based datasets. The authors created a superior dataset that combines news headlines from a mock news website and a legitimate news website in order to overcome these difficulties. They put forth a novel hybrid neural network design that, in terms of classification accuracy, outperforms current models. The research contained an attention module to clarify the model's decision-making process, highlighting the value of interpretability. The authors also make recommendations for future research, which will help to increase sarcasm detection in NLP. These recommendations include an ablation study, transfer learning on the Semeval dataset, and the incorporation of common knowledge.

Kolchinski and Potts (2018) discussed the complex relationship between author characteristics and sarcasm recognition in textual data. In order to capture author-specific nuances in sarcasm expression, the study offers two approaches, Bayesian modelling and dense embeddings, combined within a bidirectional RNN architecture. The research highlighted the effectiveness of these approaches in capturing distinct author trends using the Self-Annotated Reddit Corpus (SARC), demonstrating the usefulness of the Bayesian approach in smaller forums and the dense embedding technique in bigger, more diversified datasets. The study emphasises how crucial it is to take into account author-specific traits while performing language comprehension tasks, and it recommends additional investigation of computational tools to improve our knowledge of user behaviour and contextual language analysis.

Hazarika et al. (2018) illustrated the state-of-the-art in automated sarcasm recognition in online social media conversations. The paper makes the case that sarcasm frequently

depends on contextual assumptions and background information, offering a considerable difficulty for detection, despite the fact that existing research has primarily concentrated on lexical, syntactic, and semantic analysis. CASCADE fills this gap by combining context-driven modelling of discourse from discussion threads with user embeddings that include stylometric and personality traits. In comparison to other models, CASCADE outperforms them by incorporating both content and contextual data, especially when it comes to accurately capturing the subtle nature of sarcasm in online forums. The critical role that context and user behaviour play in strengthening sarcasm detection, CASCADE is positioned as a trailblazing hybrid model at the top of this developing study field.

Ahuja et al. (2018) gave a summary of earlier work on the subject of sarcasm detection in text data, highlighting the various methodology and approaches employed by various researchers. It covers a wide range of research topics, including the use of various classifiers and feature sets, the value of contextual information, and neural network-based models. Notably, it highlights how crucial it is to continue learning about users' psychological and behavioural makeup in order to improve the automatic detection of sarcasm. This also demonstrated the Gradient Boosting superior performance as a classifier and the persistent effectiveness of ensemble approaches over single classifier algorithms. The paper emphasises the potential significance of extending sarcasm detection to other social media platforms and expanding feature sets.

Agarwal and An (2018) developed a unique model called Affective Word Embeddings for Sarcasm (AWES) to tackle the difficult task of sarcasm detection in text. They emphasise the importance of affective content in sarcasm detection. They investigated how well affective information may be incorporated into word representations. The authors conducted a thorough analysis of several text domains and discovered that emotion-aware representations are better suited for lengthier texts (such product reviews and forum postings), whereas sentiment-aware representations excel at short text sarcasm detection (such as tweets). By presenting a data-driven strategy for enhancing sarcasm identification and illuminating the importance of emotive word embeddings in tackling this difficult problem, this work makes a contribution to the field.

Subramanian et al. (2019) emphasised the widespread use of text-based communication in social media while also recognizing its shortcomings in expressing emotional cues and the frequent occurrence of ambiguity, particularly when sarcasm is involved. It emphasises how important emojis are as a tool for improving communication and communicating emotions. The majority of current research concentrates on text-based sarcasm detection, with little attention paid to the possible insights provided by emojis. In order to improve the precision of sarcasm detection in social media posts, it highlights the need for a complete framework that incorporates both text and emoji signals. It also highlights the need of comprehending the underlying messages and intentions of users.

Liu et al. (2019) emphasised the difficulty of sarcasm recognition without physical signals by introducing the A2Text-Net, a unique deep neural network for text sarcasm detection. The A2Text-Net methodology surpassed conventional machine learning and deep learning techniques by using auxiliary variables like punctuation and part of speech. In addition to highlighting the importance of sarcasm detection in industries like social media, branding, and customer service, the study also provided a flexible framework that might be modified to operate with various deep learning models. This made a substantial contribution to improving our comprehension of natural language and provided insightful information for sentiment analysis studies in the future.

Castro et al. (2019) introduced the Multimodal Sarcasm Detection Dataset (MUSARD) and argues in favour of including multimodal cues to improve the classification of sarcasm. The research stresses the importance of multimodal fusion, underlines the shortcomings of unimodal approaches, and emphasises the demand for sophisticated spatiotemporal fusion techniques. The importance of modelling multi-party discussions and taking into account conversation-specific aspects for context modelling in sarcasm detection is also highlighted. In order to solve issues with dataset size, overfitting, and subtle contextual analysis, future research should consider the prospect of adding speaker localization and cutting-edge neural approaches. This work provides a valuable resource for future investigations into the topic of multimodal sarcasm detection by utilising instances from the MUSARD dataset.

Cai et al. (2019) tackled the increased demand for sarcasm detection in multi-modal social media messaging, particularly on Twitter, where users can mix text with graphics. The authors offer a novel multi-modal hierarchical fusion model that

combines text features, image features, and image attributes. Earlier research mainly concentrated on text-based sarcasm detection. The authors show the value of using picture characteristics as a link between text and images by implementing a fusion technique that improves the representation of each modality. A new dataset for multi-modal Twitter sarcasm detection is introduced in the paper, which also quantitatively emphasises the importance of each modality. The findings highlight the need of taking into account multi-modal information for precise sarcasm detection and suggest potential directions for further research, such as including extra common sense to the detection model and including modalities like audio.

Ren et al. (2020) proposed Sentiment Semantics Enhanced Multi-level Memory Network. They focused on Sentiments semantics as many deep learning models have not fully focused on it, but it is very important to know about sentiment semantics for better performance. They developed a multi-level memory network using sentiment semantics to catch the features of sarcasm detection. To capture sentiment semantics they used first level memory network, second level memory network is used to record the contrast between situation in each sentence and sentiment semantics. They used improved CNN to improve the memory network in the absence of local information.

Potamias et al. (2020) developed a transformer-based approach to irony and sarcasm detection. Figurative language (FL) is present all over the internet on all the social media platforms. Short texts come with greater challenges as it is difficult to understand their literal meaning due to its contrasting and metaphorical content; this remains the unsolved issue in the field of Natural Language Processing. They proposed neural network technology that builds on recently proposed pre-trained transformer-based network architecture which is further enhanced with the employment and devise of a recurrent convolutional neural network. They kept data preprocessing as minimum. They presented the first transformer-based methodology, supporting the pre-trained RoBERTa model combined with a recurrent convolutional neural network, to handle fanciful language in social media. They also wanted to cut down on preprocessing and engineered feature extraction steps which are, according to them, unnecessary when overly trained deep learning methods such as transformers are used.

Yaghoobian et al. (2021) focused on automatically finding sarcasm in written text. Sarcasm detection means figuring out when someone is saying the opposite of what

they mean in sentences that express feelings or opinions. In the history of sarcasm research, three major methods stand out: Hashtag-Driven Learning, Semi-Supervised Pattern Finding and Using Extra Information. The paper also talks about a rule-based approach, where computers look for specific signs of sarcasm using preset rules. The rules rely on finding words or ideas that have hidden emotions and label them as "positive" or "negative". For studying sarcasm, three types of text are used: short text with just one sentence, long text with sarcastic sentences mixed in with regular ones, and transcripts of conversations (TV Shows). Deep learning techniques were also used, where computers learn more about the people writing the text and the topics they discuss to get better at spotting sarcasm.

Razali et al. (2021) focused on spotting sarcasm in tweets using a combination of deep learning features and carefully crafted features related to the tweet's context. They extracted a feature set from a Convolutional Neural Network (CNN) and blended it with handcrafted features to find the best combination. They tested various machine learning techniques to classify these features and found that Logistic Regression worked best for identifying sarcasm. They used five different feature sets: deep features, incongruity features, hyperbolic features, temporality features, and dislike features. However, the temporality and dislike features didn't perform well because they were not very common in the dataset. The informal language used in tweets, including words related to time, events, nouns, verbs, and pronouns, made them difficult to detect. Despite this, these features still contributed significantly to sarcasm detection, and when combined with other features, they could be even more valuable.

Lou et al. (2021) explored a unique way to understand sarcasm. They create two special types of graphs for each sentence: an "affective graph" that captured emotional information and a "dependency graph" based on sentence structure. These graphs helped in identifying how words in a sentence affected each other emotionally. They introduce a framework called Affective Dependency Graph Convolutional Network (ADGCN) to leverage these graphs for sarcasm detection. The ADGCN has three main parts: Constructing Affective and Dependency Graphs: This step creates the emotional graph and a syntax-aware dependency graph for each sentence using both emotional knowledge and sentence structure. Learning Context Representation: They used bidirectional LSTMs (Bi-LSTM) to understand the context of the sentence and create

vector representations for it. Learning Graph Representation: they used multi-layer Graph Convolutional Networks (GCNs) to make sense of the emotional connections between words in the sentence for sarcasm detection.

Handoyo et al. (2021) proposed a smart way to spot sarcasm on Twitter. They used a powerful model called RoBERTa and boosted their dataset with Global Vector representations (GloVe) to improve word understanding and context. This helped them create more data and balance their dataset. They checked how well their model did on four different datasets. Two of them had tweets with specific hashtags (Ptacek and Ghosh), collected automatically. The other two datasets (iSarcasm and SemEval-18) were manually reviewed and included tweets from surveys and tweets marked as sarcastic by third-party reviewers. The interesting part was that when they added more data to the sarcastic tweets, it didn't make the model better at recognizing sarcasm, but it did help it get better at finding non-sarcastic text. They found that RoBERTa worked well for spotting sarcasm in various tasks.

Eke et al. (2021) suggested a way to spot sarcasm in text using a mix of deep learning and traditional machine learning. Sarcasm can make the tone of text seem opposite to what it really means, making it difficult for sentiment analysis. Understanding the context is a big challenge in handling such content. Three benchmark datasets were used which were provided by different researchers. They tried three different models for classifying the data: Bi-LSTM with GloVe Embedding: They used a deep learning approach with a Bidirectional LSTM (Bi-LSTM) and GloVe word embeddings to create context vectors for words. BERT Model: The second model relied on a powerful model called BERT, which is great at understanding context. Feature Fusion Model: This one combined BERT features, sentiment-related data, syntactic (language structure) information, and GloVe word embeddings with traditional machine learning. They evaluated their models using four important measures: precision, recall, accuracy, and F-measure. This research is notable for being the first to blend BERT features, word embeddings, and linguistic and sentiment-related information for sarcasm detection.

Bedi et al. (2021) researched on the Hindi-English code-mixed conversational dialog. Two major contributions were made by them: First, a dataset called MaSaC, which has content in both Hindi and English, for detecting sarcasm in a multi-modal way. Second, a new neural network model called MSH-COMICS that was good at classifying

sentences. The model used a special attention mechanism that helped it focus on small parts of a sentence at a time, making it efficient. A context layer was also added that looked at the conversation history to make better sense of the text. Sometimes, understanding sarcasm or humour depends on the context. For code-mixed content (mixing languages like Hindi and English), first it is figured out which language each word is in, then only a text can be processed more effectively for different tasks. MaSaC is a fresh dataset that combines both languages in a multi-modal setting, and then MSH-COMICS is used to classify the sentences. The tests showed that this approach with enhanced textual representation worked well and improved the performance in identifying sarcasm and humour.

AL-An et al. (2021) aimed to improve the detection of sarcasm using two techniques: LSTM and Auto-Encoder. They used a standard dataset and applied some preprocessing steps to the text. The key to their success was the Auto-Encoder. The tool took the document's information and tried to create the same information as output. This process helped the model learn the essential details that strongly influenced sarcasm. They also used Long Short-Term Memory (LSTM) to feed the embedding of each word into the model, which helped in creating an overall understanding of the document. The Auto-Encoder can dig deep into the data, uncovering hidden patterns and relationships through a process of encoding and decoding. This helped the model understand how different features are connected, improving sarcasm detection.

Baroiu and Matu (2022) conducted a systematic literature review on the evolution of automatic sarcasm detection from its inception in 2010 to the present day. They explored the interdisciplinary nature of this work, drawing from Artificial Intelligence, specifically Natural Language Processing (NLP). NLP has made significant progress over the past decade, with sentiment analysis being a key area of focus in both academia and industry. The challenge they addressed was how to handle figurative language, like sarcasm and irony, within sentiment analysis. Sarcasm, defined as a form of irony expressing a negative or critical attitude, relies heavily on contextual understanding. Sarcasm can introduce noise into data analysis, leading researchers to develop models for sarcasm detection, which has become a subfield within sentiment analysis and NLP. The paper's core aim was to provide a systematic literature review of this subfield, and offer valuable insights to NLP researchers. This review assisted researchers in staying

current with the latest advancements in sarcasm detection, aiding them in selecting appropriate approaches for their specific tasks.

Ren et al. (2023) proposed a knowledge-augmented neural network model for sarcasm detection. They investigated a knowledge-augmented neural network model that leverages the contextual information of the original text from external knowledge source, for sarcasm detection. Firstly, from the external knowledge source context is extracted for the original text. After that original text and context fetched at the initial step are processed sequentially through the embedding and encoding layers ,to automatically learn high-level semantic representation. After these steps the SoftMax layer is used to output the classification probability. Evaluating this model on Semeval-2018 Task 3 dataset, 82.79% F1 score was achieved. Experimental analysis also indicated the importance of contextual information in sarcasm detection.

METHODOLOGY

3.1 Corpus Creation:

Corpus creation is a major step in sarcasm detection because it is necessary to select the proper data for training. There are three types of datasets available on the internet viz short text, long text and other datasets according to Joshi et al. (2017). Short texts are like tweets and threads, which have length limitations set by the platforms. Long texts are like discussion forums and posts and other datasets are like text from novels or tv shows.

We will be using a “News Headlines” dataset, having abundance of sarcastic headlines which will be helpful in training and testing the model. We took reference for these datasets from kaggle. It is a dataset of 26,710 lines containing sarcastic and non-sarcastic headlines. We will be training data manually by marking them sarcastic or non-sarcastic.

Dataset is meticulously labelled and divided in sarcastic and non-sarcastic headlines. There is a total of 11,725 sarcastic and 14,985 non-sarcastic news headlines respectively. The model will be trained on the 67% of the dataset and tested on the remaining 33% of the same.

For Example:

Sarcastic sentence: *I love working on Sundays.*

Non-sarcastic sentence: *I love Fridays.*

Example of Dataset:

1. HEADLINE: mom starting to fear son's web series closest thing she will have to grandchild
SARCASM: True
2. HEADLINE: j.k. rowling wishes snape happy birthday in the most magical way
SARCASM: False

3.2 Extraction of Linguistic and Cognitive Features:

Linguistic features refer to the characteristic or element of the language that can be analysed and studied in communication. These features are essential for understanding how language works and how it is used for various purposes. Linguistic factors play a crucial role in sarcasm detection. Sarcasm is a form of figurative language where the intended meaning is opposite to the literal meaning of the words used. They are used to identify the cues and markers that may indicate the presence of sarcasm in a statement. Some key linguistic factors on we will be working on, are word choice, tone of voice, contextual incongruity, hyperbole, negation, situation awareness, emoji, punctuation, repetition and echoing. Cognitive function is a broad term that refers to mental processes involved in the acquisition of knowledge, manipulation of information, and reasoning. These features are central to human cognition and have significant implications for psychology, education, and other fields. Cognitive features are relevant in sarcasm detection, as understanding sarcasm often requires more than just linguistic analysis; it involves grasping the speaker's or writer's intended meaning and the underlying cognitive processes. Some cognitive features that play an important role in sarcasm detection are contextual awareness, emotion recognition, emotional intelligence, common sense reasoning, incongruity Detection and theory of mind. Sarcasm detection often combines linguistic and cognitive features, as well as machine learning techniques, to make accurate assessments. For instance, machine learning models can be trained on large datasets to recognize patterns in language and context that indicate sarcasm, and cognitive elements such as theory of mind and context comprehension can be integrated into these models to improve their accuracy.

3.3 Deep Learning Model:

Sarcasm is really frequent in both print and electronic media, but it's confusing, even for computers. People regularly use sarcasm in newspaper headlines to draw attention, but many readers miss the joke and get the false information about the news. These misunderstanding spreads when they tell others about it. So, we made an application that can spot sarcasm using LSTM. We trained it using different kinds of sentences from news and social media. Our system works well and can tell if a sentence is sarcastic or not with high accuracy.

The landscape of sarcasm detection research is diverse, showcasing various methodologies and approaches. Riloff et al. (2013) ^[32] focused on contrasting positive sentiment with negative situations to identify sarcasm, while Maynard et al. (2013) ^[23] integrated textual and multimedia data for social media analysis, emphasising structured preservation and unique mining algorithms. Ptacek et al. (2014) ^[27] explored sarcasm detection in English and Czech tweets using machine learning, addressing language complexities. Nagwanshi and Madhavan (2014) ^[25] introduced a semantic-based approach for sarcasm detection, achieving notable accuracy. Liu et al. (2014) ^[21] delved into sarcasm nuances in English and Chinese, presenting an ensemble learning method for superior performance. Rajadesingan et al. (2015) ^[28] developed SCUBA, a framework integrating user behaviour for Twitter sarcasm detection, demonstrating effectiveness with limited data. Kunneman et al. (2015) ^[19] identified hashtags as effective markers for sarcasm in Dutch tweets, highlighting the challenge of distinguishing sarcastic from literal messages. Bamman and Smith (2015) ^[5] emphasised the contextual nature of sarcasm, leveraging non-linguistic context data for improved detection accuracy. Zhang et al. (2016) ^[35] employed neural networks to detect sarcasm in tweets, outperforming traditional methods. Bouazizi and Ohtsuki (2016) ^[9] proposed a pattern-based approach for sarcasm detection on Twitter, leveraging sentiment analysis for accurate classification. Amir et al. (2016) ^[4] introduced a deep neural network that leveraged user embeddings and lexical cues for sarcasm identification, reducing the reliance on manual feature engineering. Khodak et al. (2017) ^[17] introduced the SARC corpus, a substantial dataset for sarcasm research, while Joshi et al. (2017) ^[16] presented a survey of computational approaches to sarcasm detection, emphasising the complexity of the task and the need for diverse methodologies.

Deep learning, a subset of machine learning, demonstrates superior performance when dealing with data that lacks a predefined structure. Deep learning enables computational models to perform categorization tasks using the supplied text, voice, or images while continuously learning from the given data. Deep learning models can achieve cutting-edge precision that occasionally even surpasses that of a human being. Neural network topologies with numerous layers and enormous amounts of labelled data are used to train models. However, recurrent neural networks (RNNs), convolutional neural networks (CNNs), and transformer architecture-based deep learning models, in

particular, have shown promise in collecting the contextual information and patterns required for comprehending sarcasm.

Despite the promise of these deep learning techniques, sarcasm recognition is still an unsolved research issue. Even with well-developed deep learning models, it can be difficult to create models that can comprehend and interpret humour, context, and linguistic nuance. Additionally, as adequately labelled ironic data is frequently scarce and arbitrary, gathering and labelling datasets for training such algorithms can be a challenge.

With the progress made in deep learning techniques, distinctly Long Short-Term Memory (LSTM) networks, which have shown better performance in sequence modelling tasks. It's reasonable to conclude that LSTM-based models could provide enhanced sarcasm detection potential. These models can productively capture long-range dependencies and contextual nuances in text, making them adequate for tasks like sarcasm detection, which repeatedly require understanding indistinct linguistic cues and context. By leveraging the power of LSTM networks, future research in sarcasm detection could potentially achieve even higher levels of accuracy and robustness, further advancing our understanding and capabilities in this domain.

3.4 LSTM Model:

Long Short -Term Memory Network is a powerful in AI and deep learning, was created by Hochreiter and Schmidhuber, a type of RNN architecture that addresses the problems like vanishing or exploding gradient problem and allows learning of long-term dependencies in sequential data, which makes them appropriate for tasks such as speech recognition, language translation, and time series forecasting.

The working of LSTM can be understood as, while watching a movie or reading a book sometimes we feel a sense of similarity that we have seen or read in some previous scenes or past chapters. So, similarly we can use this feature for sarcasm detection, if we feed a sentence that is sarcastic while training the model, that information will be remembered for the future use and to predict the nature of sentences while testing the model.

Given below, Fig.1 explains the internal working and the major components of an LSTM cell.

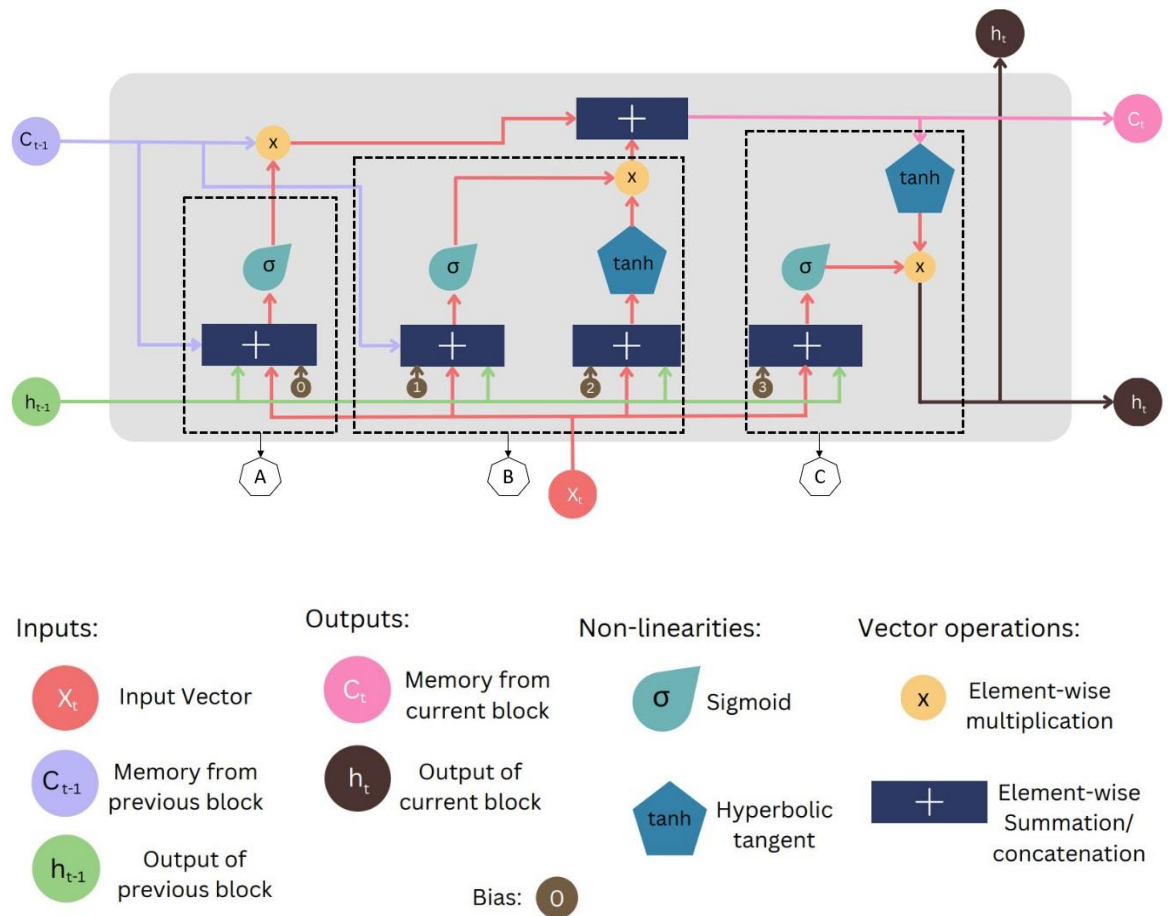


Fig. 1: LSTM Cell and its components

Our LSTM model is divided into three gates that are forget gate, input gate and output gate. The forget gate decides whether the information received from the previous cell is to be remembered or is inapt. In the second step the cell tries to learn new information for input to this cell. And at last, the updated information from the current cell state.

LSTM consists of 4 layers that interact with each other to produce output of the current cell along with cell state. And these 2 things are passed onto the next hidden layer unlike RNN. It consists of 3 logistic sigmoid gates and tanh layer.

Gates are basically used to limit the information that is passed through the cell. There are 3 types of gates present in LSTM:

A) Forget Gate:

Information that is not required anymore by the cell state is removed with a forget gate. X_t and h_{t-1} are fed to the gate and multiplied with weighted matrices followed by addition of bias. And the resultant is passed through an activation function that gives binary output.

B) Input Gate:

Now, the addition of useful information is done by the input gate. First Information is regulated using a sigmoid function similar to forget gate, then vector is created using the tanh function that gives output from -1 to 1 values of vector. And regulated values are multiplied to obtain useful information.

C) Output Gate:

The task of obtaining useful information from the current cell state to be presented as output is done by output gate. Firstly, tanh function on a cell creates a vector and after that information is regulated using sigmoid function and filtered by values to be remembered, then the value of vector and regulated vector are multiplied to obtain output of current cell and input for another block.

Cell state vector is updated by forgetting information (from forget gate) and addition of information (from input gate).

3.5 Workflow:

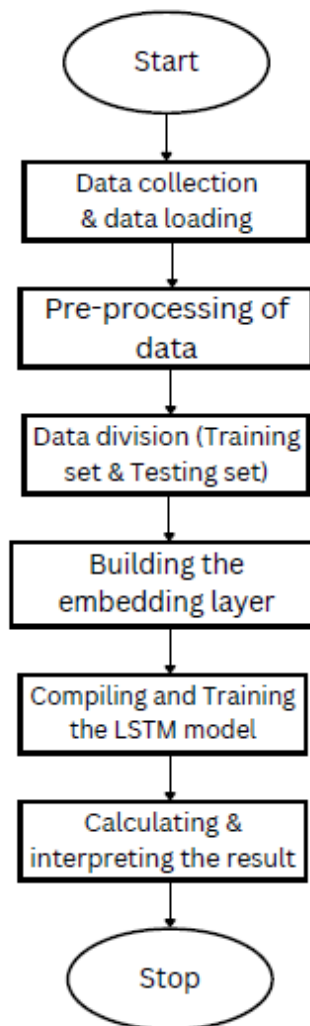
To achieve our objectives, we utilized the methodology represented in Fig. 2.

We start with data collection and loading of all the necessary data. For implementing our model, we install and use TensorFlow. TensorFlow is a rich system for managing all aspects of a machine learning system. Then, we clean the dataset by removing duplicates, handling missing values, standardizing formats and correcting errors. We also use tokenizer to break down a sequence of text into smaller, meaningful units called tokens. Then the punctuations and stop words are removed. The preprocessing of data

is then done to improve the quality, usability, and efficiency of the data for the intended task or analysis. Data is then divided into appropriate subsets for effective analysis for training and testing sets.

Fig. 2: LSTM Cell and its components

The core of our approach involved embedding the LSTM model using keras function,



in which we check if the word is present or not in our vocabulary and then their vector representation is taken in consideration, followed by compilation and training phases for calculating the accuracy of our model. Finally, our result is interpreted. This comprehensive methodology ensured a systematic and thorough approach towards achieving our objectives.

RESULT AND DISCUSSION

The results of our study demonstrates that our model is 81.95% accurate. Here, we have used news headlines as dataset. While working with LSTM, it is important to make all the inputs in a fixed size. To achieve this objective, we pad the review sentences. As we know LSTM is better in terms of capturing the memory of sequential information better than simple RNNs. We then compiled and trained our model by fixing the hyperparameters like batch size to 32, epochs to 25 and verbose to 2. Finally, we displayed the model accuracy on test data.

4.1 Quantitative Analysis:

We have analysed our model's performance on the basis of its accuracy. Accuracy is a metric used to evaluate the performance of a classification model. It measures the percentage of correctly classified instances out of all instances. It tells us how often the model makes the correct prediction. It is calculated as the number of correct predictions divided by the total number of predictions.

$$ACC = (TP + TN) / (TP + TN + FP + FN)$$

To calculate our model's accuracy, we have fixed certain hyperparameters like epochs to 25, verbose to 2 and Batch size to 1. An epoch is one complete pass through the entire training dataset during the training process. In machine learning, training a model involves updating its parameters (weights and biases) iteratively to minimize the loss function. Each epoch consists of multiple iterations, where the model sees different batches of data. Training for multiple epochs allows the model to learn from the data multiple times, potentially improving its performance. Verbose is a parameter that controls the amount of information printed during the training process. In many Machine Learning frameworks, including TensorFlow and Keras, setting verbose to 1 prints progress bars or logs during training, showing metrics such as loss and accuracy after each epoch. Setting it to 0 suppresses these outputs. Batch size refers to the number of training examples utilized in one iteration. In stochastic gradient descent (SGD), the model parameters are updated after processing each individual example. However, processing one example at a time can be inefficient due to frequent updates. Batch training aims to strike a balance between the efficiency of processing one example at a

time and the stability of processing the entire dataset at once. It divides the training data into batches, and the model parameters are updated after processing each batch. The size of the batch can significantly impact training speed and the quality of the final model. You can adjust parameters like batch size, learning rate, and the number of epochs based on the observed training and validation performance.

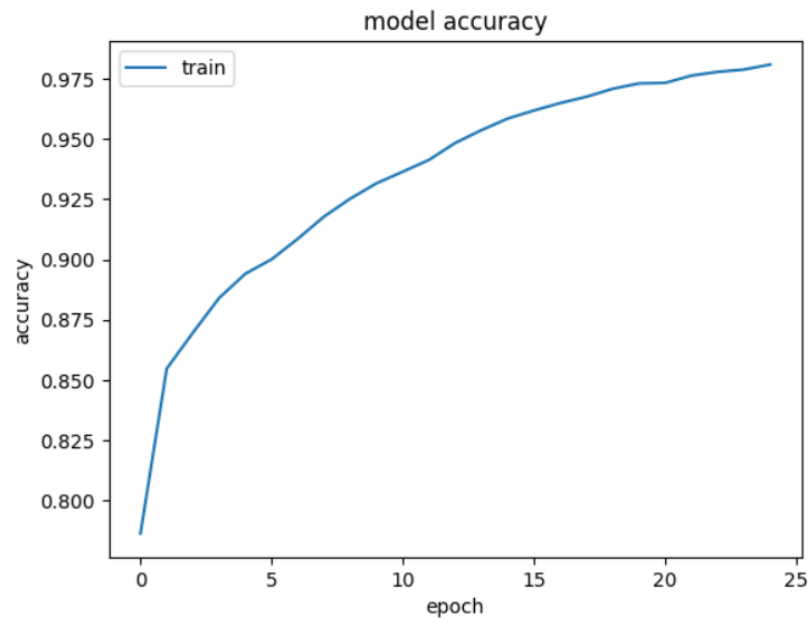


Fig. 3: Model's accuracy while training

The graph in Fig. 3 illustrates the accuracy of our model, with the x-axis denoting number of iterations and the y-axis representing accuracy. The blue line signifies the training phase. Initially, the accuracy tends to be low as the model's parameters are random and uninformative. As training progresses, the accuracy generally increases, indicating the model is learning from the training data. We can see the progressive increase of the training accuracy over epochs, indicating effective model learning. Efficiency for the training dataset increased steeply over the 1st epoch. Additionally, it provides insights into the effectiveness of the chosen model architecture and hyperparameters.

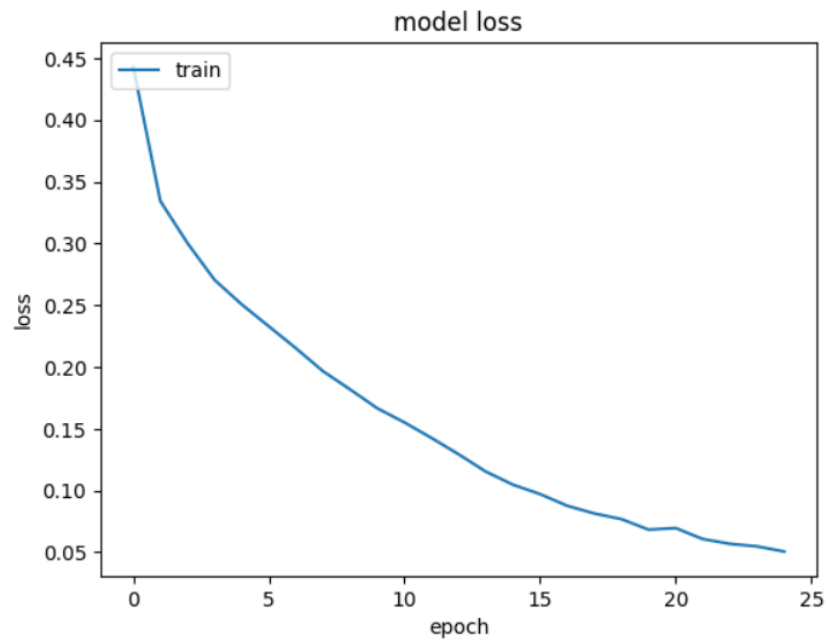


Fig. 4: Model's loss while training

The graph in Fig. 4 is depicting the model's loss during the training phase illustrates the loss function's value on the Y-axis against the number of epochs on the X-axis. Initially high, the loss steadily decreases as the model iteratively adjusts its parameters to minimize prediction errors. As training progresses, the loss tends to converge towards a minimum, indicating that the model has learned to make more accurate predictions. Monitoring this curve is vital for assessing the model's training progress and ensuring convergence. Ultimately, understanding the model's loss dynamics is essential for optimizing its performance and ensuring robust training outcomes.

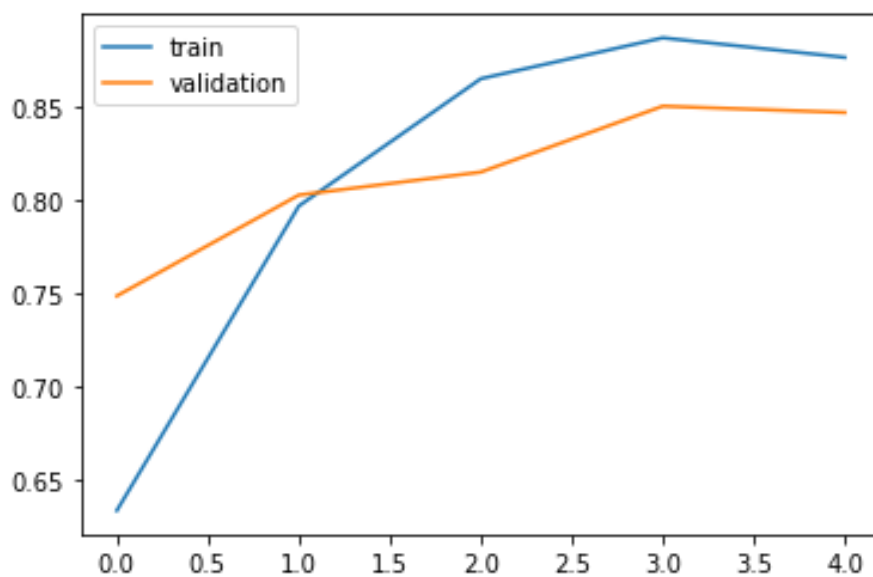


Fig. 5: Training and Testing accuracy of the model

The graph in Fig. 5 illustrates the accuracy of our model, with the x-axis denoting number of iterations and the y-axis representing accuracy. The blue line signifies the training phase, while the red line represents testing. Notably, both training and testing accuracies progressively increase over epochs, indicating effective model learning. Efficiency for the training dataset increased steeply over the 1st epoch, while there was a gradual increase for the testing phase. The minimal disparity between training and testing accuracies suggests the absence of overfitting, reflecting the robustness of our approach.

Research Paper	Authors	Model Used	Accuracy
A contextual word embedding for Arabic sarcasm detection with random forests.	Elagbry et. al. (2021)	BERT Model	73%
CASCADE: Contextual Sarcasm Detection in Online Discussion Forums	Hazarika et. al. (2018)	CUE-CNN	77%
Tweet Sarcasm Detection Using Deep Neural Network	Zhang et. al. (2016)	Deep Neural Network	78.55%
Sarcasm Detection using Contextual Word Embedding with Gaussian Model for Irony Type Identification	Krishnan et. al. (2022)	ELMO	70%
Affective Dependency Graph for Sarcasm Detection	Lou et. al. (2021)	ADCGN	72.25%
Sarcasm Detection using LSTM	Sethi et. al. (2024)	LSTM	81.95%

Table 1: Past Models along with their accuracies

Upon comparing our work with other models, it becomes evident that our model has outperformed the above-mentioned ones. We checked different things like how often it made the right predictions and how good it was at finding the right answers when given choices. Our model has done better because it is also good at understanding tricky patterns in the data. Also, we might have used some smart techniques to make our model work even better, like making sure it doesn't get too caught up in the details or combining different models together. This shows that our model is really good at

solving the problem we're working on, and it could be really useful in the real world. Plus, we made sure to test it a lot to be sure it works well in different situations.

4.2 Qualitative Analysis:

- **Drawbacks:**

Our model at times struggles with detecting sarcasm, leading to comparatively lower accuracy rates than other models.

For Example:

1. Statement: mom starting to fear son's web series closest thing she will have to grandchild

This is a sarcastic sentence from our dataset but our model detects it as a non-sarcastic sentence.

2. Statement: can't make a simple cup of coffee without everyone freaking out

This is also a sarcastic sentence from our dataset but our model detects it as a non-sarcastic sentence.

- **Efficiency:**

While our model may have failed in detecting a few sarcastic sentences, it successfully identified the majority of sentences with accuracy.

For Example:

1. Statement: whale regrets eating 290,000 plastic poker chips that fell off container ship

This is a sarcastic sentence from our dataset and our model also detects it as a non-sarcastic sentence.

2. Statement: right to own handheld device that shoots deadly metal pellets at high-speed worth all of this.

This is a sarcastic sentence from our dataset and our model also detects it as a non-sarcastic sentence.

CONCLUSION

Sarcasm plays a crucial role in communication, but spotting it is challenging even for humans, let alone computers. Newspapers often use sarcasm in headlines to grab attention, but readers often miss the irony, leading to misunderstandings. Hence, there's a growing need for systems that can reliably detect sarcasm. To address this, we've developed sarcasm detectors using neural networks, analysing how computers learn sarcasm patterns. Our project inputs sequences labelled as sarcastic or non-sarcastic, drawn from news headlines and social media commentary datasets. Through rigorous evaluation, our classifiers achieve high accuracy in distinguishing sarcasm from non-sarcasm. This work addresses the complexity of sarcasm in social media discourse, offering a promising solution to improve sentiment analysis and enhance understanding of communication dynamics.

Sarcasm detection research has evolved significantly over the past decade, encompassing a wide array of methodologies and strategies. Beginning with seminal works such as Riloff et al. (2013) ^[32], who focused on discerning sarcasm by juxtaposing positive sentiment with negative situations, and Maynard et al. (2013) ^[23], who integrated textual and multimedia data for social media analysis, scholars have explored various approaches to tackle this intricate task. Ptacek et al. (2014) ^[27] delved into sarcasm detection across different languages using machine learning, while Nagwanshi and Madhavan (2014) ^[25] introduced a semantic-based approach that achieved notable accuracy. Liu et al. (2014) ^[21] investigated sarcasm nuances in multiple languages and proposed an ensemble learning method demonstrating superior performance. Subsequent studies like Rajadesingan et al. (2015) ^[28] developed frameworks incorporating user behaviour for Twitter sarcasm detection, showcasing effectiveness even with limited data. Others, such as Kunneman et al. (2015) ^[19], emphasised the challenge of distinguishing sarcastic from literal messages, highlighting hashtags as effective indicators for sarcasm. Bamman and Smith (2015) ^[5] stressed the importance of contextual understanding, leveraging non-linguistic context data to enhance accuracy.

Further advancements came with the utilisation of neural networks by Zhang et al. (2016) ^[35] to detect sarcasm in tweets, surpassing traditional methods in performance,

and the proposal of pattern-based approaches by Bouazizi and Ohtsuki (2016) ^[9], which incorporated sentiment analysis for improved classification accuracy. Amir et al. (2016) ^[4] introduced deep neural networks utilising user embeddings and lexical cues for sarcasm identification, reducing the need for manual feature engineering. The introduction of significant datasets like the SARC corpus by Khodak et al. (2017) ^[17] and comprehensive surveys of computational approaches by Joshi et al. (2017) ^[16] marked further milestones in the field. Over this period, from 2017 to 2023, sarcasm detection research has witnessed rapid progress, with scholars emphasising the importance of contextual comprehension, linguistic nuances, and computational advancements. Incorporating diverse datasets, innovative feature sets, and multimodal cues, attention to user-specific traits, and recent advancements such as knowledge augmentation in neural network models underscore the potential for even more effective sarcasm detection algorithms with applications in sentiment analysis, social media analysis, and beyond.

Sarcasm detection research between 2017 and 2023 has seen significant strides, with researchers exploring diverse methodologies and strategies. Chaudhari and Chandankhede (2017) ^[12] investigated various techniques ranging from rule-based methods to deep learning approaches, emphasising precision and semantic depth. Agarwal and An (2018) ^[1] introduced affective word embeddings, enhancing sarcasm identification, especially in longer texts. Castro et al. (2019) ^[11] stressed the importance of multimodal cues, introducing the MUsTARD dataset and sophisticated fusion techniques for sarcasm detection. Cai et al. (2019) ^[10] addressed multi-modal sarcasm detection, combining text and image features for improved representation. Kolchinski and Potts (2018) focused on capturing author-specific nuances, emphasising the role of author traits in sarcasm expression. Handoyo et al. (2021) ^[14] utilised advanced models like RoBERTa and GloVe embeddings for sarcasm detection in code-mixed dialogues. Lastly, Ren et al. (2023) ^[31] proposed a knowledge-augmented neural network, leveraging external sources for improved contextual understanding and classification accuracy. These advancements underscore the interdisciplinary nature of sarcasm detection research and its potential applications in various domains.

The methodology employed for sarcasm detection encompasses several key stages, including corpus creation, feature extraction, and model development using deep

learning techniques, particularly LSTM architecture. Firstly, corpus creation involves compiling a diverse dataset of textual statements containing both sarcastic and non-sarcastic instances. Our dataset consists of 26,710 lines containing sarcastic and non-sarcastic headlines. There is a total of 11,725 sarcastic and 14,985 non-sarcastic news headlines respectively. This dataset is crucial for training and evaluating the sarcasm detection model effectively.

Before training the sarcasm detection model, the dataset undergoes preprocessing steps such as tokenization, lowercasing, removal of stop words, punctuation, and special characters. Additionally, techniques such as stemming or lemmatization may be applied to normalise the text. The corpus should be large enough to capture various linguistic patterns and contextual cues indicative of sarcasm, ensuring the model's robustness and generalisation capabilities. Next, extracting linguistic and cognitive features from the textual data plays a pivotal role in encoding relevant information for sarcasm detection. Linguistic features capture syntactic and semantic information from the text, which can help in identifying sarcasm. These features may include n-grams, part-of-speech tags, syntactic parse trees, sentiment scores, and discourse markers. Cognitive features focus on the cognitive processes involved in sarcasm comprehension, such as theory of mind and empathy. These features can be derived from linguistic cues that signal sarcasm, such as incongruity, hyperbole, and irony.

By incorporating a rich set of features, the model can learn to recognize subtle linguistic cues characteristic of sarcastic expressions. Deep learning techniques, particularly LSTM models, offer an effective approach for sarcasm detection due to their ability to capture long-range dependencies and temporal dynamics in sequential data. The LSTM architecture, with its gated memory units, enables the model to retain and selectively utilise contextual information, thereby enhancing its understanding of sarcasm within the given text. The LSTM model is trained on the prepared dataset, where the textual inputs are encoded along with the extracted linguistic and cognitive features. During training, the model learns to map the input sequences to their corresponding sarcasm labels through backpropagation and optimization algorithms, minimising the prediction errors and maximising detection accuracy.

Once the LSTM model demonstrates satisfactory performance in sarcasm detection, it can be deployed for various applications. The deployed model can process incoming textual data in real-time, identifying instances of sarcasm and providing appropriate responses or actions based on the detected sentiment. Continuous monitoring and updating of the model may be necessary to adapt to evolving linguistic patterns and user behaviours in different domains.

The workflow for building an LSTM model for sarcasm detection starts by defining the problem statement. Set up your development environment with necessary libraries like TensorFlow, Keras, or PyTorch. For Data Collection and Data Loading we need to Identify sources of data containing examples of sarcastic and non-sarcastic text. This could include social media platforms, news articles, or specialised datasets. Ensure that the dataset is labelled, indicating whether each example is sarcastic or not, then load the dataset into your development environment using appropriate tools or libraries like Pandas in Python. This involves reading the dataset from its storage format (e.g., CSV, JSON) into a data structure that can be manipulated and processed.

For Data Preprocessing firstly clean the text data by removing any HTML tags, special characters, punctuation, or non-alphanumeric characters that are not relevant to the analysis. Normalise the text by converting it to lowercase to ensure consistency in tokenization. Tokenize the text into individual words or subwords using techniques like whitespace tokenization or more advanced tokenizers like the NLTK library in Python. Then Convert the text tokens into numerical representations using word embeddings. This involves mapping each word to a dense vector representation in a high-dimensional space. Pre-trained embeddings like Word2Vec or GloVe can be used, or embeddings can be trained alongside the model. Pad or truncate the sequences of tokens to a fixed length to ensure that all input sequences have the same length. This is necessary for feeding the data into the LSTM model.

Data Division involves splitting the dataset into training, validation, and test sets. The training set is used to train the model, the validation set is used to tune hyperparameters and monitor the model's performance during training, and the test set is used to evaluate the final performance of the trained model. Shuffle the data to ensure that the model does not learn any spurious patterns based on the order of the examples in the dataset.

The Building of the Embedding Layer involves creation of an embedding layer as the first layer of the LSTM model. This layer is responsible for converting the input text tokens into dense vector representations. Specify the input dimension of the embedding layer to match the size of the vocabulary, which is the total number of unique words or tokens in the dataset; also specify the output dimension of the embedding layer, which determines the size of the embedding vectors.

Model Building involves defining the LSTM model architecture. This typically involves stacking one or more LSTM layers along with other layers like Dropout to prevent overfitting. Configure the LSTM layers by specifying parameters such as the number of units (i.e., hidden neurons) in each layer, the activation function (usually 'tanh'), and whether the layers should return sequences or only the final output. Then compile the model by specifying the loss function, optimizer, and metrics. For sarcasm detection, binary cross-entropy is commonly used as the loss function, Adam or RMSprop can be used as optimizers, and metrics like accuracy can be used to monitor the model's performance. Train the LSTM model using the training data. This involves passing the training data through the model in mini-batches and adjusting the model's weights based on the calculated loss. Monitor the model's performance on the validation set during training to detect overfitting or underfitting. Early stopping can be used to halt training if the performance on the validation set does not improve for a certain number of epochs. Experiment with different hyperparameters such as learning rate, batch size, number of epochs, and LSTM layer configurations to optimise the model's performance.

Finally evaluate the trained model using the test set to obtain its accuracy. Analyse any misclassifications or errors made by the model to identify patterns or characteristics of sarcastic text that the model may have difficulty capturing. Interpret the results to draw conclusions about the effectiveness of the LSTM model for sarcasm detection. This may involve comparing the model's performance to baseline models or other approaches to sarcasm detection. Based on the evaluation results, decide whether further iterations are necessary to improve the model's performance. If the model's performance is satisfactory, finalise the model for deployment or further analysis. If not, iterate on the model architecture, hyperparameters, or preprocessing steps to improve performance.

The evaluation of sarcasm detection using LSTM models reveals encouraging results across various performance metrics, underscoring the model's adeptness at discerning sarcastic utterances from non-sarcastic ones. The accuracy of our model is 81.95% that provides a comprehensive assessment of the model's efficiency in this task, showcasing its ability to correctly identify and classify instances of sarcasm with high confidence. By leveraging the inherent capacity of LSTM architectures to capture long-range dependencies and contextual information, the model excels in understanding the intricate linguistic nuances characteristic of sarcastic expressions. Here, we have used new headlines as our dataset. Our dataset consists of 26,710 lines containing sarcastic and non-sarcastic headlines. There is a total of 11,725 sarcastic and 14,985 non-sarcastic news headlines respectively.

While working with LSTM, it is important to make all the inputs in a fixed size. To achieve this objective, we pad the headlines. As we know LSTM is better in terms of capturing the memory of sequential information better than simple RNNs. We then compiled and trained our model by fixing the hyperparameters like batch size to 32, epochs to 25 and verbose to 2.

Nevertheless, challenges persist, notably regarding the quality and quantity of the training dataset, which can impede the model's performance. Additionally, the subtle and context-dependent nature of sarcasm poses difficulties in accurately capturing its linguistic cues, necessitating further research into model architectures and incorporating additional features to enhance detection accuracy. Despite these challenges, the LSTM model demonstrates robust generalisation capabilities, exhibiting consistent performance when tested on unseen data, thereby highlighting its potential for deployment in real-world applications.

Moving forward, avenues for future research may entail exploring alternative neural network architectures to improve sarcasm detection accuracy further. The successful implementation of LSTM-based sarcasm detection holds significant implications across various domains, including sentiment analysis in social media, customer feedback analysis, and conversational AI systems, where understanding subtle nuances in language is paramount for accurate interpretation and response.

APPENDICES

Accuracy	measure of the correctness of predictions made by a model
Ambiguity	inexactness
Annotation	to add a short explanation or opinion to a text or image.
Archivist	an information professional who manages and maintains important records and archives for long-term accessibility and preservation.
Batch size	number of training examples utilized in one iteration.
BERT	open-source machine learning framework for NLP.
ELMo	deep contextualized word in vectors or embeddings.
Cognitive	relating to or involving the processes of thinking and reasoning
Contextual Embeddings	vector representation of words that accurately represent their meaning within the context of their use.
Contextual incongruity	when the literal meaning of picture is different from its context.
Convolutional neural networks	type of Artificial neural network used for image recognition and processing.
Corpora	the task of converting a natural language statement to a logical form
Accrue	to increase in a number or amount
Corpus creation	complex task of collecting all the machine-readable texts.
Decipher	convert into normal language

Decision trees Supervised learning algorithm that models' decisions based on input features.

Deep Learning type of machine learning that employs neural networks with numerous layers to model and comprehend intricate data.

Deep neural network Machine learning technique similar to Artificial Neural Network.

Double entendre A word or phrase which is open to two interpretations.

Rhetorical Effective speaking or writing.

Efficiency how quickly a model can be trained or how well it performs with limited computational resources.

Embeddings High-dimensional vectors into a lower-dimensional space for easier analysis and comparison.

Emotive relating to emotions

Ensemble together

Entropy randomness

Epoch refers to one complete pass through the entire training dataset during the training of a neural network or any other machine learning model.

Explicit stated clearly and in detail

Feature extraction process of transforming raw data into a format that is suitable for input to a machine learning model.

Figurative language when words are not used for their literal meaning.

Gradient boosting machine learning ensemble technique that combines the predictions of multiple weak learners, typically decision trees, sequentially.

Heuristics a method of problem solving.

Hyperbole exaggeration to make something more exciting or more dangerous.

Incongruity something that does not fit in its location.

Interlocutors one who takes part in Conversation.

Kaggle Online Community to find datasets.

Keras Keras is a high-level neural networks API, written in Python and capable of running on top of TensorFlow, Theano, or Microsoft Cognitive Toolkit.

Knowledge-augmented model that recognize named entities in an entirely unsupervised way by using entity type information latent in the model.

Lemmatization process of reducing words to their base or canonical form (lemma), typically by removing inflections and variations, to normalise the text for analysis or processing.

Leverage using something to achieving something new and better.

Lexical cues words or phrases within text that provide hints or clues about the meaning, sentiment, or context of the surrounding content.

Lexicons contains information (semantic, grammatical) about individual word of word string

Linguistic features characteristics or properties of language, such as syntax, semantics, morphology, or phonology, that are used to analyse or understand text.

Meticulously being very precise.

Multimodal cues information derived from multiple modalities such as text, images, audio, or video, which are used collectively to infer meaning or context.

Naïve Bayes algorithm used for classification problem

NLP branch of artificial intelligence that enables computers to comprehend, interpret, and generate human language.

Novel approach fresh or innovative way of doing something, often characterized by originality, creativity, or uniqueness.

Overfitting occurs when a model learns the detail and noise in the training data to the extent that it negatively impacts its performance on unseen data

Pertinent relevant and on-point

Pragmatic dealing with things sensibly and realistically

Precision fact of being exact and accurate

Recurrent neural networks type of artificial neural network which uses sequential data or time series data

Robust The structural system's capacity to withstand significant damage under extreme loads.

SARC Semantic Analysis of Rejected Content. It refers to the process of analysing and understanding the meaning or intent behind content that has been flagged or rejected, often in the context of content moderation or filtering.

Semantic relating to meaning in language or logic.

Sentiment a view or opinion that is held or expressed.

Sentiment analysis process of analyzing digital text to determine if the emotional tone of the message is positive, negative, or neutral.

Skew when not distributed uniformly

Spatiotemporal belonging to both space and time or to space–time.

Stemming process of reducing words to their root or base form by removing suffixes or prefixes.

Stochastic Gradient Descent optimization algorithm used in training machine learning models, particularly neural networks.

Stylometric application of the study of linguistic style, usually to written language

Support Vector Machine supervised machine learning algorithm for classification and regression.

Syntactic of or according to syntax

TensorFlow open-source machine learning framework developed by Google for building and training machine learning models, particularly neural networks.

Time series forecasting creating assumptions and interpretation about a given data.

Tokenization process of breaking down a piece of text into smaller units called tokens, which could be words, phrases, or symbols.

Trailblazing the activity of introducing new ideas or methods.

Transcends to rise above or go beyond the limits.

Unimodal involving one mode

Verbose refers to a mode of operation where the system provides detailed output or information, often for debugging or monitoring purposes.

Word embedding technique in natural language processing where words or phrases from a vocabulary are mapped to vectors of real numbers in a continuous vector space, preserving semantic relationships between words.

REFERENCES

1. Agrawal, A., & An, A. (2018, June). Affective representations for sarcasm detection. In The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval (pp. 1029-1032).
2. Ahuja, R., Bansal, S., Prakash, S., Venkataraman, K., & Banga, A. (2018). Comparative study of different sarcasm detection algorithms based on behavioural approach. *Procedia computer science*, 143, 411-418.
3. AL-Ani, M. M., Omar, N., & Nafea, A. A. (2021). A Hybrid Method of Long Short-Term Memory and Auto-Encoder Architectures for Sarcasm Detection. *J. Comput. Sci*, 17(11), 1093-1098.
4. Amir, S., Wallace, B. C., Lyu, H., & Silva, P. C. M. J. (2016). Modelling context with user embeddings for sarcasm detection in social media. *arXiv preprint arXiv:1607.00976*.
5. Bamman, D., & Smith, N. (2015). Contextualized sarcasm detection on twitter. In *proceedings of the international AAAI conference on web and social media* (Vol. 9, No. 1, pp. 574-577).
6. Băroiu, A. C., & Trăușan-Matu, Ș. (2022). Automatic sarcasm detection: Systematic literature review. *Information*, 13(8), 399.
7. Bedi, M., Kumar, S., Akhtar, M. S., & Chakraborty, T. (2021). Multi-modal sarcasm detection and humor classification in code-mixed conversations. *IEEE Transactions on Affective Computing*.
8. Bharti, S. K., Pradhan, R., Babu, K. S., & Jena, S. K. (2017). Sarcasm analysis on twitter data using machine learning approaches. *Trends in Social Network Analysis: Information Propagation, User Behavior Modeling, Forecasting, and Vulnerability Assessment*, 51-76.
9. Bouazizi, M., & Ohtsuki, T. O. (2016). A pattern-based approach for sarcasm detection on twitter. *IEEE Access*, 4, 5477-5488.
10. Cai, Y., Cai, H., & Wan, X. (2019, July). Multi-modal sarcasm detection in twitter with hierarchical fusion model. In *Proceedings of the 57th annual meeting of the association for computational linguistics* (pp. 2506-2515).
11. Castro, S., Hazarika, D., Pérez-Rosas, V., Zimmermann, R., Mihalcea, R., & Poria, S. (2019). Towards multimodal sarcasm detection (an _obviously_ perfect paper). *arXiv preprint arXiv:1906.01815*.

12. Chaudhari, P., & Chandankhede, C. (2017, March). Literature survey of sarcasm detection. In 2017 International conference on wireless communications, signal processing and networking (WiSPNET) (pp. 2041-2046). IEEE.
13. Eke, C. I., Norman, A. A., & Shuib, L. (2021). Context-based feature technique for sarcasm identification in benchmark datasets using deep learning and BERT model. *IEEE Access*, 9, 48501-48518.
14. Handoyo, A. T., & Suhartono, D. (2021). Sarcasm detection in Twitter--performance impact while using data augmentation: Word embeddings. *arXiv preprint arXiv:2108.09924*.
15. Hazarika, D., Poria, S., Gorantla, S., Cambria, E., Zimmermann, R., & Mihalcea, R. (2018). Cascade: Contextual sarcasm detection in online discussion forums. *arXiv preprint arXiv:1805.06413*.
16. Joshi, A., Bhattacharyya, P., & Carman, M. J. (2017). Automatic sarcasm detection: A survey. *ACM Computing Surveys (CSUR)*, 50(5), 1-22.
17. Khodak, M., Saunshi, N., & Vodrahalli, K. (2017). A large self-annotated corpus for sarcasm. *arXiv preprint arXiv:1704.05579*.
18. Kolchinski, Y. A., & Potts, C. (2018). Representing social media users for sarcasm detection. *arXiv preprint arXiv:1808.08470*.
19. Kunneman, F., Liebrecht, C., Van Mulken, M., & Van den Bosch, A. (2015). Signalling sarcasm: From hyperbole to hashtag. *Information Processing & Management*, 51(4), 500-509.
20. Liu, L., Priestley, J. L., Zhou, Y., Ray, H. E., & Han, M. (2019, December). A2text-net: A novel deep neural network for sarcasm detection. In 2019 IEEE first international conference on cognitive machine intelligence (CogMI) (pp. 118-126). IEEE.
21. Liu, P., Chen, W., Ou, G., Wang, T., Yang, D., & Lei, K. (2014). Sarcasm detection in social media based on imbalanced classification. In *Web-Age Information Management: 15th International Conference, WAIM 2014, Macau, China, June 16-18, 2014. Proceedings 15* (pp. 459-471). Springer International Publishing.
22. Lou, C., Liang, B., Gui, L., He, Y., Dang, Y., & Xu, R. (2021, July). Affective dependency graph for sarcasm detection. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 1844-1849).

23. Maynard, D., Dupplaw, D., & Hare, J. (2013). Multimodal sentiment analysis of social media.
24. Misra, R., & Arora, P. (2019). Sarcasm detection using hybrid neural networks. arXiv preprint arXiv:1908.07414.
25. Nagwanshi, P., & Madhavan, C. V. (2014, October). Sarcasm detection using sentiment and semantic features. In *International Conference on Knowledge Discovery and Information Retrieval* (Vol. 2, pp. 418-424). SCITEPRESS.
26. Potamias, R. A., Siolas, G., & Stafylopatis, A. G. (2020). A transformer-based approach to irony and sarcasm detection. *Neural Computing and Applications*, 32, 17309-17320.
27. Ptáček, T., Habernal, I., & Hong, J. (2014, August). Sarcasm detection on Czech and English twitter. In *Proceedings of COLING 2014, the 25th international conference on computational linguistics: Technical papers* (pp. 213-223).
28. Rajadesingan, A., Zafarani, R., & Liu, H. (2015, February). Sarcasm detection on twitter: A behavioural modelling approach. In *Proceedings of the eighth ACM international conference on web search and data mining* (pp. 97-106).
29. Razali, M. S., Halin, A. A., Ye, L., Doraisamy, S., & Norowi, N. M. (2021). Sarcasm detection using deep learning with contextual features. *IEEE Access*, 9, 68609-68618.
30. Ren, L., Xu, B., Lin, H., Liu, X., & Yang, L. (2020). Sarcasm detection with sentiment semantics enhanced a multi-level memory network. *Neurocomputing*, 401, 320-326.
31. Ren, Y., Wang, Z., Peng, Q., & Ji, D. (2023). A knowledge-augmented neural network model for sarcasm detection. *Information Processing & Management*, 60(6), 103521.
32. Riloff, E., Qadir, A., Surve, P., De Silva, L., Gilbert, N., & Huang, R. (2013, October). Sarcasm as contrast between a positive sentiment and negative situation. In *Proceedings of the 2013 conference on empirical methods in natural language processing* (pp. 704-714).
33. Subramanian, J., Sridharan, V., Shu, K., & Liu, H. (2019). Exploiting emojis for sarcasm detection. In *Social, Cultural, and Behavioral Modeling: 12th International Conference, SBP-BRiMS 2019, Washington, DC, USA, July 9–12, 2019, Proceedings 12* (pp. 70-80). Springer International Publishing.

34. Yaghoobian, H., Arabnia, H. R., & Rasheed, K. (2021). Sarcasm detection: A comparative study. arXiv preprint arXiv:2107.02276.
35. Zhang, M., Zhang, Y., & Fu, G. (2016, December). Tweet sarcasm detection using deep neural networks. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: technical papers* (pp. 2449-2460).

LIST OF PUBLICATIONS

1. **Sethi A., Sharma A., Gautam A.S., Sakshi** (2024) presented a poster on “Sarcasm Detection using LSTM”.

26th Annual Conference of the Society of Statistics, Computer and Applications (SSCA), organised by Department of Mathematics and Statistics in affiliation with Banasthali Vidyapith, Rajasthan on **Emerging Trends of Statistical Sciences in AI and its Applications (ETSSAA-2024)**.

2. **Sethi A., Sharma A., Gautam A.S., Sakshi** (2024) published a paper on “Sarcasm Detection using LSTM”.

9th Internation Conference on Information, Communication and Computing Technology (ICICCT-2024) by Jagan Institute of Management Studies (JIMS), New Delhi. {Communicated}



Sarcasm Detection using Contextual Embeddings

Aahna Sethi¹, Akanksha Sharma¹, Ananya Singh Gautam¹, Sakshi¹, Nisheeth Joshi^{1,2}, Pragma Katyayan^{1,2}

¹Department of Computer Science, Banasthali Vidyapeeth, Rajasthan.

²Speech and Language Processing Lab, Centre for Artificial Intelligence, Banasthali Vidyapeeth, Rajasthan.

nisheeth.joshi@rediffmail.com, pragma.katyayan21@gmail.com, aahnasethi18@gmail.com,

aakankshash1311@gmail.com, 78ananyasingh@gmail.com, pandeysakshi9203@gmail.com

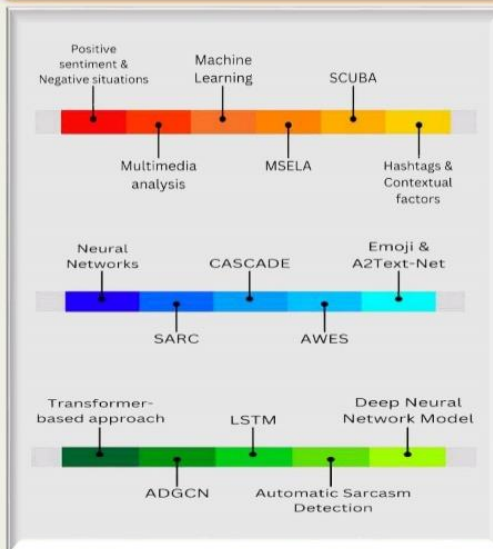
Abstract

Sarcasm detection, is an important step for sentiment analysis because it is not easy to tell the sentence is sarcastic or non-sarcastic unlike other sentiment. And also, it is crucial for machine to detect sarcasm for better understanding to serve as an interface for mutual communication between machines and humans. Sarcasm detection is still an unexplored part because not only for machine but sometimes humans are also not able to understand sarcasm. In this study we have tried to understand sarcasm and train the model using LSTM.

Introduction

- Sarcasm is the cutting use of words, often with ambivalence and frequently in a playful way, to make fun of someone or something.
- Sarcastic detection is crucial for understanding online interactions, sentiment analysis, and chatbot performance, as figurative language, including irony and sarcasm, serves different purposes.
- Adapting to various contextual scenarios, the proposed model uses LSTM networks and contextual embeddings to improve sarcasm detection.

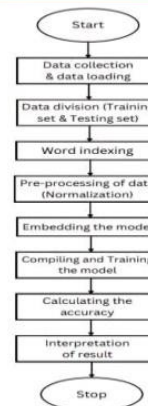
Literature Review



Scope

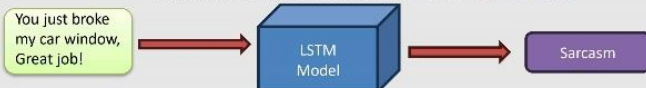
- This study investigated the detection efficacy of sarcasm using contextual embeddings, tackling the problem of context-dependency and subtlety in NLP.
- LSTM networks were utilized to capture semantic subtleties and temporal dynamics.
- Empirically explored the synergistic effects of embeddings and LSTMs, leading to improved architectural configurations and training methodologies.
- Advanced knowledge of sequential reasoning and contextual comprehension.

Methodology



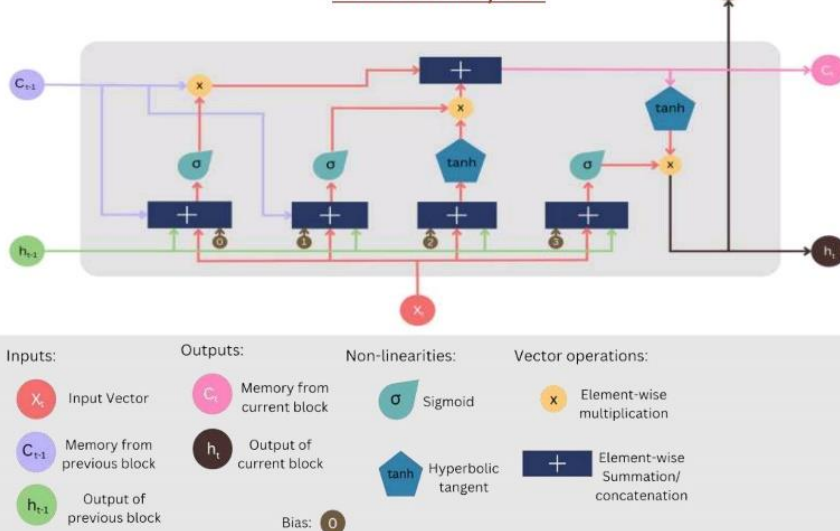
Goal

To obtain the whether the sentence is sarcastic or non-sarcastic

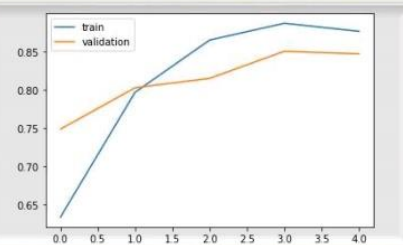


Processing Technique

LSTM Memory Cell



Performance Analysis



Conclusion

- This poster shows our performed experiment using LSTM. In this a brief description of our model is also explained.
- We have mentioned the past works done in this domain and also mentioned about our goal. Later, methodology and processing techniques are also discussed.
- We have mentioned the model accuracy on test data

References

- Bamman, D., & Smith, N. (2015). Contextualized sarcasm detection on twitter. In proceedings of the international AAAI conference on web and social media (Vol. 9, No. 1, pp. 574-577).
- Bharti, S. K., Pradhan, R., Babu, K. S., & Jena, S. K. (2017). Sarcasm analysis on twitter data using machine learning approaches. Trends in Social Network Analysis: Information Propagation, User Behavior Modeling, Forecasting, and Vulnerability Assessment, 51-76.
- Cai, Y., Cai, H., & Wan, X. (2019, July). Multi-modal sarcasm detection in twitter with hierarchical fusion model. In Proceedings of the 57th annual meeting of the association for computational linguistics (pp. 2506-2515).
- Joshi, A., Bhattacharya, P., & Carman, M. J. (2017). Automatic sarcasm detection: A survey. ACM Computing Surveys (CSUR), 50(5), 1-22.
- Katyayan, P., & Joshi, N. (2020). Sarcasm Detection Algorithms Based on Sentiment Strength. *Intelligent Data Analysis: From Data Gathering to Data Comprehension*, 289-306.
- Liu, P., Chen, W., Ou, G., Wang, T., Yang, D., & Lei, K. (2014). Sarcasm detection in social media based on imbalanced classification. In Web-Age Information Management: 15th International Conference, WAIM 2014, Macau, China, June 16-18, 2014. Proceedings 15 (pp. 459-471). Springer International Publishing.
- Potamias, R. A., Siolas, G., & Stafylidis, A. G. (2020). A transformer-based approach to irony and sarcasm detection. *Neural Computing and Applications*, 32, 17309-17320.
- Zhang, M., Zhang, Y., & Fu, G. (2016, December). Tweet sarcasm detection using deep neural networks. In Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: technical papers (pp. 2449-2460).

In a Nutshell

- Identified the range of reviews.
- Padded the sequences to ensure they are of a fixed size.
- Fixed hyperparameters such as batch size, epochs, and verbose.
- Compiled and trained the model using the specified hyperparameters..
- Finally, evaluated the model's accuracy on the test data.

81.95%
Accuracy