Detecting Sarcasm Across Headlines and Text

A Project Report submitted in the partial fulfilment

of the Requirements for the award of the degree

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

Submitted by

A. Lakshmi Niharika (21471A0573)

S. Neelima (21471A05B8)

K. Nikhitha (21471A0590)

Under the Esteemed Guidance of

Shaik Rafi, M. Tech.,(Ph.D.,)

Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE

AND ENGINEERING

NARASARAOPETA ENGINEERING COLLEGE: NARASAROPET

(AUTONOMOUS)

Accredited by NAAC with A+ Grade and NBA under Tyre -1 NIRF rank in the band of 201- 300 and an ISO 9001:2015 Certified

Approved by AICTE, New Delhi, Permanently Affiliated to JNTUK, Kakinada

KOTAPPAKONDAROAD, YALAMANDAVILLAGE, NARASARAOPET- 522601

2024-2025

NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project that is entitled with the name "**Detecting Sarcasm Across Headlines and Text**" is a bonafide work done by the team **A. Lakshmi Niharika (21471A0573), S. Neelima (21471A05B8), K. Nikhitha (21471A0590)** in partial fulfilment of the requirements for the award of the degree of bachelor of technology in the Department of computer science and engineering during 2024-2025.

PROJECT GUIDE

Shaik Rafi, M.Tech.,(Ph.D.,)

Assistant Professor

PROJECT CO-ORDINATOR

Dr. M. Sireesha, M.Tech.,Ph.D.,

Associate Professor

HEAD OF THE DEPARTMENT

Dr. S. N. Tirumala Rao, M.Tech., Ph.D.,

Professor & HOD

EXTERNALEXAMINER

DECLARATION

We declare that this project work titled "**Detecting Sarcasm Across Headlines and Text**" is composed by ourselves that the work contain here is our own except where explicitly stated otherwise in the text and that this work has not been submitted for any other degree or professional qualification except as specified.

A. Lakshmi Niharika (21471A0573)

S. Neelima (21471A05B8)

K. Nikhitha (21471A0590)

ACKNOWLEDGEMENT

We wish to express our thanks to various personalities who are responsible for the completion of the project. We are extremely thankful to our beloved chairman sri **M. V. Koteswara Rao**, B.Sc., who took keen interest in us in every effort throughout this course. We owe out sincere gratitude to our beloved principal **Dr. S. Venkateswarlu**, M.Tech., Ph.D., for showing his kind attention and valuable guidance throughout the course.

We express our deep felt gratitude towards **Dr. S. N. Tirumala Rao**, M.Tech., Ph.D., HOD of CSE department and also to our guide **Shaik Rafi**, M.Tech., (Ph.D.,) Assisatnt professor of CSE department whose valuable guidance and unstinting encouragement enable us to accomplish our project successfully in time.

We extend our sincere thanks towards **Dr. M.Sireesha**, M.Tech, Ph.D., Associate professor & Project coordinator of the project for extending her encouragement. Their profound knowledge and willingness have been a constant source of inspiration for us throughout this project work.

We extend our sincere thanks to all other teaching and non-teaching staff of department for their cooperation and encouragement during our B.Tech degree.

We have no words to acknowledge the warm affection, constant inspiration and encouragement that we received from our parents.

We affectionately acknowledge the encouragement received from our friends and those who involved in giving valuable suggestions had clarifying out doubts which had really helped us in successfully completing our project.

By

A. Lakshmi Niharika (21471A0573)

S. Neelima(21471A05B8)

K. Nikhitha(21471A0590)

NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

INSTITUTE VISION AND MISSION

INSTITUTION VISION

To emerge as a Centre of excellence in technical education with a blend of effective student centric teaching learning practices as well as research for the transformation of lives and community.

INSTITUTION MISSION

M1: Provide the best class infra-structure to explore the field of engineering and research.

M2: Build a passionate and a determined team of faculty with student centric teaching, imbibing experiential, innovative skills.

M3: Imbibe lifelong learning skills, entrepreneurial skills and ethical values in students for addressing societal problems.

NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

VISION OF THE DEPARTMENT

To become a center of excellence in nurturing the quality Computer Science & Engineering professionals embedded with software knowledge, aptitude for research and ethical values to cater to the needs of industry and society.

MISSION OF THE DEPARTMENT

The department of Computer Science and Engineering is committed to

M1: Mould the students to become Software Professionals, Researchers and Entrepreneurs by providing advanced laboratories.

M2: Impart high quality professional training to get expertize in modern software tools and technologies to cater to the real time requirements of the Industry.

M3: Inculcate team work and lifelong learning among students with a sense of societal and ethical responsibilities.

NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

Program Specific Outcomes (PSO's)

PSO1: Apply mathematical and scientific skills in numerous areas of Computer Science and Engineering to design and develop software-based systems.

PSO2: Acquaint module knowledge on emerging trends of the modern era in Computer Science and Engineering

PSO3: Promote novel applications that meet the needs of entrepreneur, environmental and social issues.

Program Educational Objectives (PEO's)

The graduates of the programme are able to:

PEO1: Apply the knowledge of Mathematics, Science and Engineering fundamentals to identify and solve Computer Science and Engineering problems.

PEO2: Use various software tools and technologies to solve problems related to academia, industry and society.

PEO3: Work with ethical and moral values in the multi-disciplinary teams and can communicate effectively among team members with continuous learning.

PEO4: Pursue higher studies and develop their career in software industry.

NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

Program Outcomes

Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

Problem analysis: Identify, formulate, research literature, and analyse complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

Modern tool usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practices.

Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

Project Course Outcomes (CO'S):

CO421.1: Analyze the System of Examinations and identify the problem.

CO421.2: Identify and classify the requirements.

CO421.3: Review the Related Literature.

CO421.4: Design and Modularize the project.

CO421.5: Construct, Integrate, Test and Implement the Project.

CO421.6: Prepare the project Documentation and present the Report using appropriate method.

Course Outcomes – Program Outcomes mapping

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO 2	PSO3
C421.1		✓											✓		
C421.2	✓		✓		✓								✓		
C421.3				✓		✓	✓	✓					✓		
C421.4			✓			✓	✓	✓					✓	✓	
C421.5					✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
C421.6									✓	✓	✓		✓	✓	

Course Outcomes – Program Outcome correlation

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
C421.1	2	3											2		
C421.2			2		3								2		
C421.3				2		2	3	3					2		
C421.4			2			1	1	2					3	2	
C421.5					3	3	3	2	3	2	2	1	3	2	1
C421.6									3	2	1		2	3	

Note: The values in the above table represent the level of correlation between CO's and PO's:

1. Low level
2. Medium level
3. High level

Project mapping with various courses of Curriculum with Attained PO's:

Name of the Course from Which Principles Are Applied in This Project	Description of the Task	Attained PO
C2204.2, C22L3.2	Defining the problem and applying Deep learning techniques for predict the Sarcastic text & headlines.	PO1, PO3
CC421.1, C2204.3, C22L3.2	Critically analyzed project requirements and identifying suitable process models for experiments.	PO2, PO3
CC421.2, C2204.2, C22L3.3	Logical design is done by using the unified modelling language which involves team work.	PO3, PO5, PO9
CC421.3, C2204.3, C22L3.2	Each and every model is tested, integrated, and evaluate the models in our project.	PO1, PO5
CC421.4, C2204.4, C22L3.2	Documenting experiments, results, and findings collaboratively within the group.	PO10
CC421.5, C2204.2, C22L3.3	Presenting each phase of the project, including raw data analysis and evaluation, in a group Periodically.	PO10, PO11
C2202.2, C2203.3, C1206.3, C3204.3, C4110.2	Implementing and validating models, Project will be handled by the Detecting Sarcastic text future updates.	PO4, PO7
C32SC4.3	Designing a web interface to visualize predictions and verify model evaluation metrics effectively.	PO5, PO6

ABSRTACT

In this era with the rapid growth in social media usage among the current generation, a huge amount of content and comments, most of them sarcastic, is seen. Sarcasm has turned out to be an important part of daily life, especially in news and social media, where sarcastic comments are often used for better attention. However, detecting sarcasm is always challenging because it deals with understanding the difference between what has been said and what is meant. The current paper focuses on the detection of sarcasm in news headlines with the help of deep learning. Previous works were based on a wide range of datasets. However, these had limitations regarding either size or quality. In this respect, the authors propose creating a new dataset of headlines from sarcastic news sites and real news sites that is large and of high quality, hence appropriate for machine learning model training. The authors have also used the CNN-BILSTM architecture for text analysis, identifying sarcasm expression and deciding whether it is sarcastic or not-sarcastic which gained an accuracy of 97%. This dataset is made publicly available to enable further research in this direction.

INDEX

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

These days, the phone and technology are slowly and gradually becoming an inseparable part of our existence. We cannot even imagine a single day without social media browsing, video watching, or messaging. In turn, because of this, social media has become congested, and people express themselves more than ever. But with the rise of social media comes an increase in sarcastic comments and humorous postings. It would seem like each one of them had now become some kind of stand-up comedian online. Some use sarcasm to be funny or make a point, others to cover their feelings or even cruelly. Whichever way it goes, sarcastic comments are taking over in social media, and this presumably changes how we interact online.

Sarcasm is a subtle way of expressing things where the words mean quite the opposite of their literal sense. Sarcasm is a versatile form of communication that can be expressed in various ways, depending on the speaker's intent and the context of the conversation. While some types of sarcasm are lighthearted and humorous, others can be biting and critical. Understanding the different types of sarcasm helps in recognizing its use and avoiding potential misunderstandings. Below are some common types of sarcasm and their characteristics.

TYPES OF SARCSAM:

Here's a more detailed explanation of each type of sarcasm, along with additional insights and examples:

1. Self-Deprecating Sarcasm :

Self-deprecating sarcasm is when a person makes fun of themselves in a humorous way. This type of sarcasm is often used to lighten the mood, make others feel at ease, or show humility. It is commonly used by comedians and people who want to appear relatable.

Example:

"Oh sure, I'm the best cook ever! Just ignore the burnt toast."

Usage:

- Helps build social connections by making people laugh.
- Often used to downplay personal achievements or mistakes.
- Can be an effective way to handle criticism by making fun of oneself before others do.

Extra Insight: While self-deprecating sarcasm can be charming, excessive use might make a person seem insecure or overly negative about themselves.

2. Deadpan Sarcasm :

Deadpan sarcasm is delivered with a completely serious expression and tone, making it difficult for some people to recognize. It is often used in professional settings or among friends who understand each other's humor well.

Example:

A student who hasn't studied might say: "Oh yes, I'm totally prepared for this exam."

Usage:

- Effective in delivering dry humor.
- Common in TV shows and movies, especially in sarcastic characters.
- Can be confusing if the listener does not pick up on the irony.

Extra Insight: Because deadpan sarcasm lacks obvious cues like exaggerated tone or facial expressions, it can sometimes be mistaken for a serious statement.

3. Brooding Sarcasm :

Brooding sarcasm carries a bitter or resentful undertone. It is often used when someone is frustrated, annoyed, or disappointed. This type of sarcasm is not necessarily humorous but serves as a passive-aggressive way to express dissatisfaction.

Example:

Someone forced to work overtime might say: "Great, just what I needed – more work on my weekend!"

Usage:

- Used when someone feels overburdened or unappreciated.
- Can be a way to express frustration indirectly.
- Sometimes comes across as passive-aggressive rather than humorous.

Extra Insight: While brooding sarcasm can be a way to vent emotions, it can also create tension if used excessively, especially in professional or personal relationships.

4. Polite Sarcasm :

Polite sarcasm appears respectful on the surface but contains underlying mockery. This type is subtle and often used to criticize or tease without being too direct.

Example:

If someone makes a mess, another person might say: "Wow, you're so neat and

organized!"

Usage:

- Often used in formal or professional settings to deliver criticism lightly.
- Can be a tool for diplomacy saying something without being too blunt.
- Can sometimes be misunderstood as genuine praise.

Extra Insight: Polite sarcasm is common in British humor and is often used in witty, clever exchanges. However, if the listener does not detect the sarcasm, the intended message may be lost.

5. Obnoxious Sarcasm :

Obnoxious sarcasm is more aggressive and can be used to belittle or insult someone. It often involves exaggerated mockery and is sometimes meant to make the other person feel foolish.

Example:

If someone makes a mistake, another person might say: "Oh wow, you're a genius!"

Usage:

- Often used among friends in a teasing manner.
- Can be offensive if used in the wrong context.
- Common in arguments or heated discussions.

Extra Insight: Obnoxious sarcasm can damage relationships if used excessively. While some people use it for humor, others may find it rude or hurtful.

6. Manic Sarcasm :

Manic sarcasm is characterized by exaggerated enthusiasm and energy, making it obvious that the speaker is being sarcastic. This type of sarcasm is playful and often used to make light of annoying situations.

Example:

"Oh wow, I just LOVE waiting in long lines!"

Usage:

- Used to express frustration in a humorous way.
- Often found in social media posts and memes.
- Helps people cope with frustrating situations by turning them into jokes.

Extra Insight: Manic sarcasm is generally harmless and can be a way to bond with others over shared annoyances. However, if used too often, it might come across as overly negative.

It is often employed to mock or to show contempt. It is widely used while speaking and in writing with the intention of drawing attention and stating one's opinion. The detection of sarcasm remains a challenging task, as it is by nature ambiguous in nature and depends on context and common sense knowledge. Hence, even humans and machines find it difficult to correctly detect the presence of sarcasm.

USE IN DAILY LIFE:

sarcasm is commonly used in various situations, from casual conversations to media and entertainment. Here are some ways sarcasm is integrated into everyday life:

- **In Social Interactions:** People use sarcasm to add humor to conversations, tease friends, or lighten a situation.
- **At Work or School:** Sarcasm can be used to deal with stress or express mild frustration in a humorous way. Example: "Oh yes, I just LOVE last-minute deadlines!"
- **In Literature and Media:** Many books, TV shows, and movies use sarcasm to make dialogue more engaging and entertaining. Characters like Chandler Bing from Friends and Dr. House from House M.D. are known for their sarcastic wit.
- **Online and Social Media:** Sarcasm is widely used in memes, tweets, and comments to express opinions humorously or critically.

Recognizing and Understanding Sarcasm:

Understanding sarcasm requires awareness of tone, body language, and context. Since sarcasm often relies on exaggeration or contradiction, key signs include:

- A dry or exaggerated tone of voice
- Facial expressions such as raised eyebrows or smirking
- Situational irony where the statement contradicts reality

The need for sarcasm detection in NLP is immense in the present digital age, since a huge amount of content is being produced on online platforms such as social media and news websites. Most of the informal languages with contextual references end up in noisy datasets, which in turn makes the process of detection tough. Also, the labelling of sarcasm in datasets manually is a time-consuming process and may lead

to inconsistencies in many cases due to different levels of interpretation of sarcasm[1]. Research in sarcasm detection has applied several different techniques: rule-based approaches, machine learning, and deep learning. The collection of social media datasets is typically performed through tag-based supervision, which is both error-prone and limited with regard to vocabulary as mentioned. At the same time, high quality, manually labelled datasets are rather small in size and too costly to produce, which results in underpowered models. More overwhelming is the fact that models have to comprehend the subtlety and contextual dependency of sarcastic language. While NLP has come a long way, most of the models lack this subtlety in sarcasm and instead depend on lexical cues rather than actual comprehension[2]. While sarcasm detection keeps improving, new challenges have opened up for researchers in multi-modal analysis that is, considering both the visual and audio cues that would give a better understanding of the nuances of sarcastic expression. The recent trend towards voice assistants and podcasts has raised the requirement for sarcasm detection in spoken language, which introduces new challenges when dealing with audio data and prosody[4]. Sarcasm detection is also an essential component in improving the effectiveness of recommendation systems. Many online platforms, such as e-commerce websites and streaming services, rely on user- generated reviews and comments to suggest products or content[7]. However, if a sarcastic review is misinterpreted as genuine praise, it can lead to incorrect recommendations and negatively impact user experience. By integrating sarcasm detection into recommendation algorithms, platforms can refine their suggestions to be more aligned with true user sentiments.

Another area where sarcasm detection is making an impact is in political discourse analysis. Political debates and discussions, particularly on social media, often involve sarcasm to criticize opponents or express strong opinions. Automated sarcasm detection can help analyze political sentiment more accurately and provide deeper insights into public opinions, policy discussions, and election trends[8,9]. Furthermore, recent studies have explored the role of psychological and cognitive factors in sarcasm detection. Research suggests that humans rely on a combination of linguistic patterns, prior knowledge, and emotional cues to recognize sarcasm. Integrating such human-like reasoning into NLP models remains a challenge but is a promising direction for future research. Methods such as knowledge graphs and

external commonsense reasoning databases can assist models in understanding sarcasm better by incorporating facts and context. One of the persistent challenges in sarcasm detection is handling code-mixed language, where users switch between multiple languages within a single text.

The cultural and linguistic diversity of online platforms further require the models to be apt for different styles of sarcasm and idioms. In order for better algorithms that are robust and generalizable, there is a definite need for future models to develop stronger cognitive and psychological insights in designing better models that enhance capturing context-dependent inferences and implicit meaning underlying human use of sarcasm[10,11,12]. This will open up new applications in sentiment analysis, opinion mining, and human-computer interaction by really pushing the limits of sarcasm detection and further enriching our understanding of language and communication. The application of sarcasm detection extends beyond just social media and sentiment analysis. It plays a crucial role in fields such as customer service automation, mental health analysis, and fake news detection[13,14]. By accurately identifying sarcastic intent, businesses can improve chatbot interactions and sentiment analysis tools, leading to better customer engagement[15]. Similarly, in mental health applications, detecting sarcasm in user comments can help identify distress signals that might otherwise be overlooked by traditional sentiment analysis methods.

Additionally, sarcasm detection is a valuable asset in fake news detection, as misleading headlines often employ sarcasm to manipulate readers. The ability to correctly interpret sarcasm can help flag potentially misleading content, contributing to the fight against misinformation.

From a technical standpoint, advancements in deep learning models, particularly transformers such as BERT and RoBERTa, have shown significant improvements in sarcasm detection. These models leverage contextual embeddings to understand subtle nuances in text better. Moreover, ensemble learning, which combines multiple models, has been observed to enhance performance by mitigating individual model weaknesses. Future research in sarcasm detection is also exploring zero-shot and few-shot learning techniques, which enable models to generalize better across different datasets without extensive retraining. Furthermore, the incorporation of multimodal sarcasm detection, utilizing images, videos, and voice inputs, is expected to refine sarcasm detection capabilities further. Hybrid

neural networks were also experimentally proved to perform better in sarcasm detection. They focused on the relevant parts of the text. The paper ”Sarcasm detection using news headlines dataset”. These models can process sequential data effectively and are thus suitable to analyze news headlines, wherein sarcasm is often employed. The aim of this study is to conduct the task of news headline sarcasm detection using deep learning techniques that are advanced. It would leverage high-quality datasets and new neural network architectures to present an improved performance and interpretability of sarcasm detection models for an overall improved performance in sentiment analysis and natural language understanding. Here, the prime focus of our project is to make the model detect sarcasm in both headlines and general sarcastic text taking information from the abovementioned paper, to enhance the detecting capability of sarcasm by our model.

2. LITERATURE SURVEY

The authors , Chy, M. S. R., Chy, M. S. R., Mahin, M. R. H., Rahman, M. M., Hossain, M. S., Rasel, A. in ”Sarcasm Detection in News Headlines Using Evidential Deep Learning-Based LSTM and GRU,”[1] reviews the related work for various approaches to detect sarcasm. Few important works include the STSM algorithm, which fuses pragmatic and lexicon-based features and ensemble models using many word- embedding techniques. The transformer-based approaches are CNN-RoBERTa. Further, analysis of historical data using lexicon-based techniques has also been attempted. Emphasize that the integration of reliability and uncertainty measures plays an important role in sarcasm detection, addressed by Evidential deep learning.

R.Misra, P.Arora, entitled ”Sarcasm detection using news headlines dataset”[2]. The base paper reviews the related works regarding prior research in sarcasm detection, citing limitations of the existing datasets and models. The authors have emphasized the need for high-quality datasets and more sophisticated models in order to capture the subtlety of sarcasm by referring to seminal works like Amir et al. ,Sarcasm Detection in Twitter: A Deep Learning Approach, and Joshi et al.A Survey of Sarcasm Detection Techniques. They instead propose a new large- scale News Headlines Dataset, tailored for the purpose of sarcasm detection, arguing outperforming previous datasets. They have further used the Hybrid Neural Network architecture comprising CNN and LSTM components along with the attention mechanism, which they have proved effective using different analysis.

”Sarcasm Detection in News Headlines using Supervised Learning” gives an overview of several methods for sarcasm detection and brings out the importance of context. Deep neural networks and user embeddings have also been employed by researchers to give better results. For example, Amir et al. used the combination of user and word embeddings, while Hazarika et al. designed a contextual detection system by using both user and content embeddings[5]. Kolchinski et al. used a Bayesian approach by using dense embeddings to classify in social media.

The discussion of the paper ”Deep Learning for Sarcasm Identification in News Head-lines” is done. Discussed revisits sarcasm detection by presenting a number of datasets and machine learning techniques. Therefore, this research used high-quality datasets such as sarcastic headlines from The Onion, The Sarcasm Corpus V2, etc., while earlier studies had solely relied on noisy data from Twitter. This will also be discussed in-depth, since advanced techniques LSTM networks as mentioned in the paper[6] as performing abstract summarization and CNN-are efficient and seemingly brought more improvements in the accuracy of enhancements in sarcasm detection of NLP.

Sarcasm detection is a relatively new challenge in text analysis. Various methods have been used in an attempt to address it, as reflected in the paper [7]. Some of these approaches, such as LSTM and various neural networks, focus on different aspects: some work with text, others with sentences, and some with words. Some methods, like CASCADE, use mixed con-tent and context-based approaches, while SCUBA places special emphasis on emotional differences and user behaviour. Other approaches are attention-based models, BERT embedding- based methods, and some approaches based on BiLSTM and affective graph representation. Most of them incorporate context, user behaviour, or emotional knowledge to present more accuracy in sarcasm detection and report remarkable results in the field.

The problem of detecting sarcasm is deliberated in the paper ”Detecting Sarcasm in News Headlines” by Onyinye ChudiIwueze and Haithem Afli, [8]focusing on news headlines and social media content. Sarcasm detection has become one of the challenging tasks in NLP due to subtlety and resting so much on context. Different methods are debated in the paper as in [13] and [14] from different angles but with a strong support for feature extraction techniques as an indispensable block in improving model performance. It provides a framework to improve the systems on the detection of sarcasm by analyzing the different approaches that exist and giving a proposal for future directions to reach the solution. Decoding Sarcasm: Unveiling Nuances in Newspaper Headlines”[9] - a paper on challenges for sarcasm detection in NLP based on newspaper headlines very effectively situates this challenge. It is the subtle contextual cues flipping the literal meaning of words that make headline sarcasm hard to detect. This paper elaborates on the different feature extraction and modelling techniques the authors tried, setting up context to a few nuances of sarcastic speech as

same in [12]. Work acts like an eye-opener in improving the models for sarcasm detection. It opens the doors to much more accurate and advanced systems. O.Chudi-Iwueze, H.Afli, of ”Sarcasm Detection with a New CNN+BiLSTM Hybrid Neural Network and BERT Classification Model” addresses the challenge of detecting sarcasms in social media-a place where sarcastic expressions occur quite often and have mostly been misinterpreted in NLP[10]. They also propose a hybrid model that includes CNN, BiLSTM, and BERT to enhance the efficiency of sarcasm detection as similar to [11,15]. It handles the intricateness of sarcastic language through a model that provides informal, contextual communication online to reduce misunderstandings and improve the capabilities.

The paper by Saurabh Porwal, Gaurav Ostwal, Anagha Phadtare, Mohini Pandey, and Manisha V. Marathe explores sarcasm detection as a key challenge in opinion mining. Researchers have studied various properties of sarcasm, such as semantic, syntactic, and lexical features, to develop detection algorithms. This study utilizes a Recurrent Neural Network (RNN) model combined with Long Short-Term Memory (LSTM) cells implemented on TensorFlow to automatically extract features from Twitter tweets. The model captures syntactic and semantic information to improve sarcasm detection accuracy. The paper also presents the model’s results and a statistical analysis of the dataset used in the study.

The paper by Bsir Bassem, Chaima Ammamou, Seifeddine Mechti, and Rim Faiz explores cross-lingual sarcasm detection using the Sarcasm Headlines and ArSarcasm datasets[12]. It evaluates three transformer-based models BERT, DistilBERT, and RoBERTa to assess their effectiveness in detecting sarcasm across languages. Performance is measured using accuracy, precision, recall, and F1-score, with results showing up to 0.99 accuracy on English headlines and 0.84 on Arabic content. These findings highlight the strong sarcasm detection capabilities of transformer models and emphasize the impact of dataset-specific features in cross-lingual sarcasm detection, paving the way for future research in this area.

The paper by Aruna Bhat and Aditya Chauhan explores multimodal sarcasm detection, addressing the growing use of images and videos alongside text on social media. While most sarcasm detection research focuses solely on text-based inputs, this study integrates both textual and visual features to improve detection accuracy. The proposed model is based on RoBERTa (a Facebook-optimized version of BERT), enhanced with a co-attention layer to capture context incongruency between text and

image attributes. Gated Recurrent Unit (GRU) is used to extract text-based features, while Feature-wise Linearly Modulated ResNet Blocks process image features. These, combined with the CLS token from RoBERTa, form the final prediction. The model demonstrates performance improvements over existing state-of-the-art approaches when tested on a publicly available Twitter dataset.The paper by Tanya Jain, Nilesh Agrawal, Garima Goyal, and Niyati Aggrawal addresses the challenges of sarcasm detection in social media by focusing on the common sarcasm pattern of "positive sentiment attached to a negative situation"[16]. The study highlights issues such as ambiguous definitions of sarcasm, evolving language trends, and the impact of emoticons on sentiment polarity. To overcome these challenges, the authors propose two ensemble-based approaches voted ensemble classifier and random forest classifier along with a pragmatic classifier for detecting emoticon-based sarcasm. Unlike traditional sarcasm detection methods that depend on pre-existing sentiment corpora, this approach employs a seeding algorithm to generate training data. The proposed model improves text summarization and sentiment analysis by addressing sarcasm, a factor often ignored in social network analysis.

The paper by Yin Wang, Xuyang Xu, Ziteng Gao, and Xi Shi presents a DialoGPT-based sarcasm detection model that enhances recognition by incorporating sarcasm probability information about the speaker and embedded emojis in sentences. Unlike previous methods that rely on complex network designs or non-corresponding pre-training models, this approach leverages pre-training on sarcasm tasks to optimize performance. The model is tested on the Sarcasm on Reddit dataset and achieves a 77 F1-score, outperforming other methods across four key metrics while maintaining efficiency. The study highlights the potential application of this approach in education, where it can help analyze students' intentions and personalities based on their use of sarcasm.

The study by Abhilasha Sharma, Abhi Uday Pandey, and Apoorv Gupta explores deep learning models for sarcasm detection in news headlines, focusing on BiLSTM and BERT. A key novelty in their research is the use of hybrid BERT models for detecting sarcasm in non-social media text. Their findings indicate that combining BERT with GBDT (Gradient Boosting Decision Trees) outperforms other strategies, highlighting its effectiveness in enhancing AI systems that require contextual understanding of sarcasm.

The study by Aruna Bhat and Govind Narayan Jha focuses on the sarcasm detection process, comparing various machine learning and deep learning models on datasets such as Twitter, Reddit, and SARC. The research explores multiple approaches, including Support Vector Machine (SVM), Convolutional Neural Networks (CNN), and Long Short-Term Memory (LSTM) networks for sarcasm detection. The paper aims to evaluate and contrast these methods based on their accuracy and effectiveness in detecting sarcasm, highlighting advancements in NLP-based sarcasm classification.

The study by Manoj Y. Manohar and Pallavi Kulkarni presents an NLP and corpus-based approach for sarcasm detection on Twitter. The research compares data using ontology-based emotion detection to classify tweets as sarcastic or non-sarcastic. The paper highlights the significance of sentiment analysis in social networks and emphasizes how text-based features aid in detecting sarcasm, ultimately improving opinion mining and user sentiment interpretation.

Unlike traditional binary classification methods, the study introduces a regression-based approach to predict sarcasm intensity instead of classifying text as simply sarcastic or non-sarcastic. The models used in this research include machine learning and deep learning models, but specific details on the models are not provided in the abstract. The proposed approach aims to enhance sarcasm detection by considering the nuanced and subjective nature of sarcasm, and the dataset will be publicly available for future research. The literature survey highlights the diverse approaches and methodologies researchers have employed for sarcasm detection across different datasets and platforms. Traditional machine learning models such as Support Vector Machines (SVM) and Random Forest Classifiers have been used to identify sarcasm based on handcrafted linguistic and syntactic features.

Overall, advancements in sarcasm detection have led to more robust, accurate, and interpretable models, enhancing sentiment analysis, opinion mining, and AI-driven conversational systems. Future work can focus on context-aware sarcasm detection, sarcasm in dialogues, and multimodal fusion techniques to improve the understanding of sarcasm in a broader range of real-world scenarios.

3. EXISTING SYSTEM

Sarcasm has increasingly become a dominant aspect of online communication, particularly in social media and news headlines. Given the rapid growth of social media, sarcastic comments have gained traction, often used to grab attention, make a humorous point, or even mask true emotions. In many cases, sarcasm serves as a form of satire or criticism, making it an essential element of modern discourse. However, while sarcasm adds depth and expressiveness to communication, it also introduces challenges, particularly when trying to detect and interpret it accurately in text-based formats. Researchers have explored various techniques, including deep learning models, lexicon-based methods, and ensemble approaches, to improve sarcasm detection. These methods aim to bridge the gap between literal and intended meanings by leveraging linguistic patterns, sentiment analysis, and contextual understanding. Traditional approaches, such as rule-based and lexicon-based techniques, rely on predefined word lists and sentiment scores to identify sarcasm. However, these methods often struggle with more nuanced expressions of sarcasm. In contrast, machine learning and deep learning models have introduced new possibilities by training algorithms to recognize sarcastic patterns from large datasets. Transformer-based models, such as RoBERTa and BERT, have shown significant promise in detecting sarcasm by analyzing contextual relationships between words and phrases. Furthermore, evidential deep learning techniques have been developed to incorporate reliability and uncertainty measures, enhancing the robustness of sarcasm detection models. The Main Problem of Sarcasm Detection

Lack of Explicit Indicators

One of the biggest challenges in sarcasm detection is the absence of explicit indicators in text-based communication. Unlike verbal conversations, where sarcasm can be detected through tone, pitch, and facial expressions, written text lacks such cues. News headlines, in particular, are short and concise, making it even harder to determine whether they contain sarcasm.

Additionally, sarcasm often employs subtle phrasing, wordplay, and indirect implications, making it difficult for traditional models to detect it accurately. A sarcastic remark can be highly context-dependent and may not contain obvious clues

such as negative words or irony markers. This challenge underscores the necessity of advanced natural language processing (NLP) techniques that can identify subtle linguistic cues and infer sarcasm from implicit meanings.

The Need for Contextual Understanding

Sarcasm often depends on understanding the context in which a statement is made. This includes cultural references, historical events, and general background knowledge. Without this deeper understanding, it becomes difficult for AI models to identify whether a statement is sarcastic or serious. For example, consider the news headline: "The Government Finally Solves All Economic Problems. By Ignoring Them." This sentence is sarcastic, but to recognize it, one must understand the actual economic situation and why "ignoring problems" would not be a real solution. One major challenge in sarcasm detection is that sarcasm often creates a contradiction between the literal meaning of a sentence and its intended message. A model that only looks at the words may mistakenly believe that the government has indeed solved economic issues, rather than realizing that the sentence is mocking the situation. This highlights the need for context-awareness in sarcasm detection. To improve sarcasm detection, AI models must be able to process external knowledge sources, such as news reports, economic data, or public sentiment. For instance, if a model can analyze recent news about economic struggles, it can better detect the sarcasm in the given headline. Additionally, sarcasm is expressed differently across cultures and languages, which further complicates its detection. Some phrases that are sarcastic in one language may not have the same effect in another, requiring models to adapt to different ways sarcasm.

Another important aspect is historical and social context. Many sarcastic statements reference previous events, political decisions, or common beliefs within a society. AI systems must be trained to recognize these patterns to differentiate between sincere and sarcastic expressions.

By developing more context-aware and knowledge-driven sarcasm detection models, researchers can improve the accuracy of NLP applications. This would benefit various fields such as sentiment analysis, misinformation detection, and AI-driven communication systems, making them more effective in understanding human language.

Complexity of Linguistic Features

Sarcasm is a complex form of communication that includes irony, exaggeration, and contradiction. These elements make it difficult for traditional text-processing models to detect sarcasm because such models usually focus on the literal meaning of words rather than the hidden intent behind them. Many sarcasm detection models use lexicon-based techniques, which rely on predefined lists of sarcastic words or phrases. However, sarcasm is highly flexible and does not always follow fixed patterns, making these approaches ineffective in many cases.

One key feature of sarcasm is irony, where a person says something but actually means the opposite. For example, if someone says, "Oh great, another Monday!" with a frustrated tone, they are not actually excited about Monday. Similarly, exaggeration is another common sarcasm feature, where a statement amplifies reality to an extreme level. For example, saying "I waited in line for a thousand years" is an exaggerated way of expressing frustration. Simple rule-based models often struggle to recognize these expressions because they require an understanding of the underlying context. To accurately detect sarcasm, AI models must be able to understand meaning beyond individual words. This requires advanced techniques such as semantic analysis, which helps the model understand the intended meaning behind a phrase, and context-based inference, which allows the model to analyze previous statements and background knowledge. Additionally, sarcasm often involves nuanced word relationships, meaning that words may take on different meanings depending on how they are used in a sentence. By incorporating deep learning and advanced natural language processing (NLP) techniques, sarcasm detection models can become more effective. These models analyze not only words but also the structure of sentences, tone, and even historical patterns of speech. As AI continues to improve, sarcasm detection will play a crucial role in making machines better at understanding human emotions and communication styles, leading to improvements in chatbots, sentiment analysis, and content moderation systems.

Ambiguity in Language

Many sarcastic comments look like genuine statements, making it hard to tell if someone is being sarcastic or serious without extra context. This creates a big challenge for both humans and AI models trying to detect sarcasm. Ambiguity in

language adds another layer of difficulty. The same sentence can have multiple meanings depending on how it is understood. For example, consider the headline: “Best Decision Ever: Company Lays Off 1,000 Employees.” A reader who is unaware of the context might think this is a positive statement about the company’s decision. However, someone familiar with the situation might immediately recognize the sarcasm in the phrase "Best Decision Ever", which is actually criticizing the layoffs. Without deeper understanding, an AI model may struggle to determine whether the statement is sarcastic or sincere. Ambiguity becomes even more complicated when sarcasm is mixed with humor. In many cases, sarcasm is used to make a joke while also delivering criticism. This overlap between humor and sarcasm makes it difficult for AI to classify the statement correctly. A model that only looks at words and sentence structure might fail to detect sarcasm because it cannot grasp the deeper meaning, emotional undertones, or the contradiction between words and intent. To handle such challenges, sarcasm detection models must go beyond simple text analysis. They need to understand sentiment polarity (whether a statement is positive or negative), detect hidden emotions, and recognize contextual contradictions. Advanced AI models use deep learning and natural language processing (NLP) to analyze these factors. They can compare statements with past data, learn from patterns, and even use external knowledge to interpret sarcasm more accurately. Improving sarcasm detection will help in many areas, such as social media analysis, automated content moderation, and chatbots. As AI advances, sarcasm detection models will become more context-aware and emotionally intelligent, making human-computer interactions more natural and effective.

Dependence on Advanced Machine Learning Techniques

Traditional machine learning methods struggle to detect sarcasm because they often rely on surface-level text analysis. Sarcasm is complex, involving hidden meanings, contradictions, and context-dependent interpretations. Simple rule-based or lexicon-based models often fail because sarcasm does not always follow predictable patterns. To overcome these limitations, researchers now use advanced deep learning models that can understand sarcasm more effectively. Deep learning techniques, such as neural networks, transformers, and attention mechanisms, help models learn the deeper context of text. These models analyze word relationships, sentiment shifts, and contradictions within a statement to determine whether sarcasm is present.

4. PROPOSED SYSTEM

The proposed solution aims to enhance sarcasm detection in news headlines and general text using a CNN-BiLSTM deep learning model. A new dataset from sarcastic (The Onion) and real (The HuffPost) news sources ensures high-quality training data. Advanced preprocessing techniques like tokenization, stopwords removal, and lemmatization improve text consistency. The model, trained on Google Colab, achieves high accuracy, leveraging CNN for local features and BiLSTM for contextual dependencies.

In our study, we utilized two datasets for sarcasm detection. First, the headlines dataset was meticulously curated to prevent issues such as incorrect labeling and linguistic inconsistencies. Additionally, since these headlines are crafted by professional journalists, the dataset gains an added layer of credibility. Many user-generated datasets often contain problems like slang, typos, and informal language, which introduce noise and make it harder for models to perform well. However, in this case, the structured nature of the language allows the model to focus solely on sarcasm's subtle nuances without being influenced by unnecessary text variations.

Raw text data often contains unnecessary elements like punctuation marks, special characters, stopwords, and inconsistent capitalization, which can reduce the accuracy of deep learning models. To improve model performance, a structured text preprocessing pipeline is necessary to clean and refine the data. This process ensures that the model focuses on meaningful patterns rather than irrelevant variations in text. Below are the key preprocessing techniques used: Tokenization involves breaking text into smaller units called tokens, which can be words, subwords, or characters. This helps convert unstructured text into a structured format, making it easier for models to analyze and understand language patterns. Many texts contain contractions like “can’t” (cannot) or “won’t” (will not). Expanding such short forms into their full versions improves text clarity and ensures that the model learns the correct meaning of words. Stopwords are commonly used words like "the," "is," and "and" that do not add much meaning to a sentence. Removing them reduces noise in the data and helps the model focus on the important words that carry meaningful context.

Lemmatization converts words to their base or root form which can be helpful while

preserving their meaning. Unlike stemming, which cuts off word endings, lemmatization ensures that words remain grammatically correct, improving the overall text quality.Converting all text to lowercase ensures consistency and avoids treating words like “Good” and “good” as separate entities. This helps in reducing redundancy and improving the efficiency of the model. Text data often contains punctuation marks, emojis, and symbols that do not contribute to understanding sarcasm. Removing these elements cleans the dataset and ensures that the model focuses on relevant words and patterns.

- **Model Architecture: CNN-BiLSTM**

The CNN-BiLSTM model follows a structured pipeline that processes textual data, extracts meaningful patterns, and predicts whether a given sentence contains sarcasm. The architecture consists of several layers, each contributing uniquely to sarcasm detection. Initially, raw text is passed through a preprocessing module that removes noise and converts text into a structured numerical format. The preprocessed text is then embedded into a dense vector representation using word embedding techniques such as Word2Vec or GloVe. This step ensures that semantically similar words have closer vector representations, aiding the model in capturing relationships between words.

The CNN layer is responsible for extracting local features such as n-grams, which help identify sarcasm-related phrases. Convolutional filters slide over the embedded text to detect important textual patterns. The extracted features are then passed to the BiLSTM layer, which processes text bidirectionally. Unlike traditional LSTM, the BiLSTM model reads the text in both forward and backward directions, allowing it to capture contextual dependencies more effectively. This is particularly useful in sarcasm detection, where words appearing later in a sentence can completely alter its meaning. The fully connected layer further processes these extracted features and refines them before passing them to the output layer, which classifies the sentence as sarcastic or non-sarcastic.

Once the text is preprocessed, it is converted into numerical format using word embeddings. Word embeddings transform words into dense vector representations, preserving semantic relationships between words. Pre-trained embeddings such as

GloVe and Word2Vec are widely used in deep learning models due to their ability to capture word similarities. Alternatively, trainable embeddings can be used, allowing the model to learn custom word representations specific to sarcasm detection. The word embedding layer plays a crucial role in mapping textual input into a high-dimensional space where meaningful patterns can be learned by the subsequent layers.

To effectively detect sarcasm in text, our proposed model utilizes a hybrid deep learning approach that combines Convolutional Neural Networks (CNNs) and Bidirectional Long Short-Term Memory (BiLSTM) networks. This fusion enables the model to capture both local word patterns and long-range contextual dependencies, ensuring a comprehensive understanding of sarcastic text. To capture both local word features and contextual relationships in sarcasm detection, the proposed model leverages a hybrid deep learning approach combining:

- **Convolutional Neural Network (CNN):**

CNNs are widely used in computer vision but have also proven effective in natural language processing (NLP) for extracting local features and patterns in text. In sarcasm detection, CNN helps identify key n-grams (short sequences of words) that indicate sarcasm, such as specific phrases or sentiment markers.

The CNN layer is designed to extract local features from the embedded text. This layer consists of multiple convolutional filters that slide over the input text, capturing n-gram features that indicate sarcasm. CNN is particularly effective in detecting specific word patterns that contribute to sarcastic expressions, such as the combination of positive and negative words in unexpected contexts. For example, a phrase like "Oh great, another Monday" exhibits sarcasm, where "great" is typically positive, but in the given context, it conveys dissatisfaction.

Each convolutional filter applies a sliding window operation to extract features from different parts of the sentence. These filters operate at various levels, detecting patterns of different lengths, such as bi-grams and tri-grams. After the convolution operation, an activation function such as ReLU (Rectified Linear Unit) is applied to introduce non-linearity, allowing the model to learn complex patterns. The extracted feature maps are then passed through a max-pooling layer, which selects the most important features while reducing dimensionality. Pooling helps in preserving critical information while discarding redundant data, improving computational efficiency.

- **How CNN Works in the Model**

Extracts Local Patterns (n-grams)

Identifies short, recurring sequences of words that commonly appear in sarcastic text.

Example: Phrases like ″Oh, great" or "Just what I needed″ can strongly indicate sarcasm.

Detects Sentiment-Carrying Features

CNN picks up word-level sentiment clues that may indicate irony or sarcasm.

Example: A sarcastic statement may contain both positive and negative sentiment words (e.g., ″Such a wonderful disaster!").

Filters and Highlights Important Features

CNN applies filters (kernels) that detect specific sarcasm-related patterns in text.

Helps the model focus on relevant sarcastic cues while ignoring unrelated words.

- **CNN Layers Used in the Model**

Embedding Layer: Converts words into dense vector representations.

Convolutional Layers: Apply filters to capture patterns in n-grams (word sequences).

Max-Pooling Layer: Reduces the size of extracted features while keeping the most important ones.

- **Bidirectional Long Short-Term Memory (BiLSTM):**

While CNN captures local patterns, BiLSTM ensures the model understands the broader context in which sarcasm appears. Since sarcasm often relies on contextual clues (such as contradictions or tone shifts), BiLSTM helps capture long-range dependencies in text. The BiLSTM layer is responsible for capturing long-range dependencies and contextual relationships between words. Unlike standard LSTM, which processes text sequentially from left to right, BiLSTM consists of two LSTM networks—one reading the text forward and the other reading it backward. This bidirectional approach allows the model to capture dependencies that span across the sentence, ensuring that the meaning of a word is interpreted within its complete context.

For sarcasm detection, context plays a significant role. A sarcastic sentence often derives its meaning from later words in the sequence. For example, in the sentence "I just love being ignored," the sarcasm is evident only after reading the entire sentence. BiLSTM ensures that both past and future words contribute to the interpretation of

sarcasm, making it more effective than unidirectional models. The hidden states from both forward and backward LSTM networks are concatenated to provide a comprehensive representation of the text, which is then passed to the fully connected layer for further processing.

- **How BiLSTM Works in the Model**

Processes Text in Both Directions (Forward & Backward)

Unlike a standard LSTM that reads text in one direction, BiLSTM processes words from both left to right and right to left, ensuring better context understanding.

Example: In the sarcastic sentence "Oh sure, because that went so well last time!", understanding "last time" is crucial to interpreting sarcasm.

Captures Long-Term Dependencies

Some sarcastic expressions require understanding words far apart in the sentence.

Example: "Wow, I totally love waiting in long queues for hours!" (The sarcasm depends on recognizing the contradiction between "love" and "waiting in long queues for hours").

Improves Sequential Learning in Text

Enhances sarcasm detection by recognizing shifts in sentiment and tone across longer passages. By combining CNN and BiLSTM, our model benefits from both word-level feature extraction (via CNN) and contextual sequence learning (via BiLSTM). This hybrid approach ensures:CNN captures local sarcasm-indicating patterns. BiLSTM learns sarcasm-related long-range dependencies.

The model effectively detects sarcasm in both short headlines and longer general text.

This combination allows the model to analyze both word-level sarcasm indicators (CNN) and contextual sarcasm (BiLSTM) effectively. After feature extraction from CNN and sequential pattern learning from BiLSTM, the extracted features are passed to a fully connected (dense) layer which is shown in Fig: 4.1. This layer consists of multiple neurons that process and refine the features before classification. The fully connected layer serves as a bridge between feature extraction and prediction, ensuring that the learned representations are transformed into meaningful outputs.

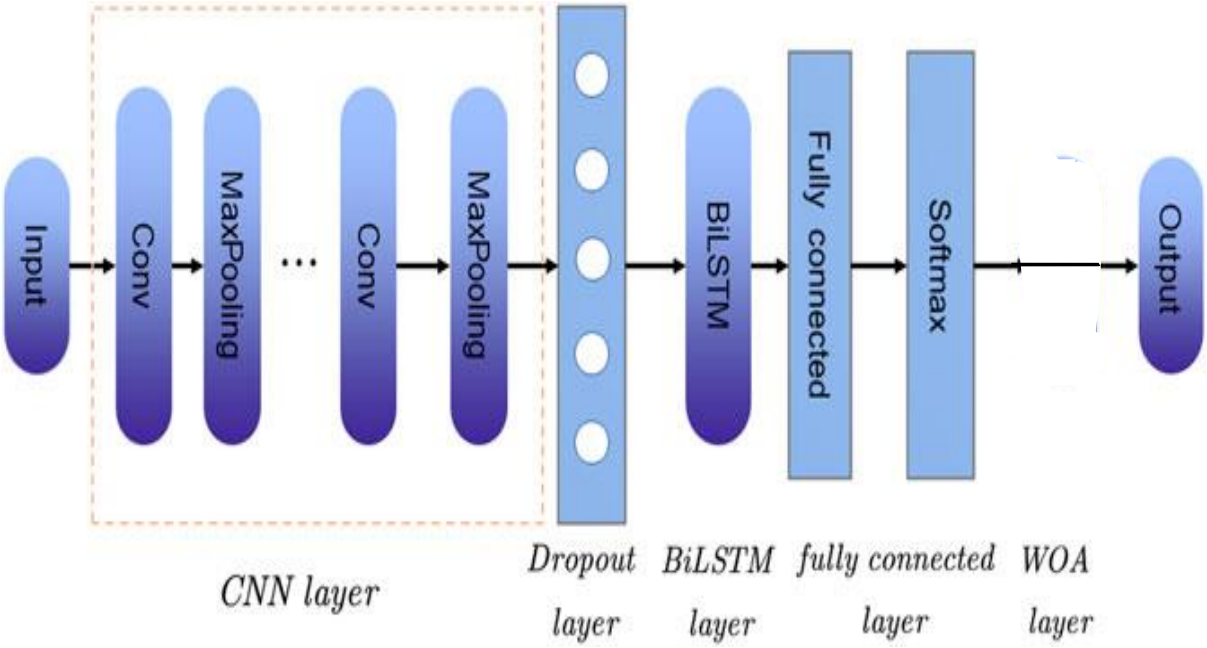

Fig : 4.1 Model Architecture

The given Fig: 4.1 diagram represents a hybrid deep learning model combining CNN (Convolutional Neural Networks) and BiLSTM (Bidirectional Long Short-Term Memory) for processing sequential data, such as text. The model begins with an input layer, where the data (such as text embeddings) is fed into the network. The CNN layer, consisting of convolutional and max-pooling operations, extracts local features from the input while reducing dimensionality. After feature extraction, a dropout layer is applied to prevent overfitting by randomly deactivating some neurons during training. The processed features are then passed to the BiLSTM layer, which captures long-range dependencies in both forward and backward directions, enhancing contextual understanding. The extracted features are further transformed by a fully connected layer, followed by a softmax layer, which converts the output into probability scores for classification. Finally, the output layer provides the classification result, making the model suitable for tasks such as sarcasm detection, sentiment analysis, and text classification.To prevent overfitting, dropout regularization is applied, randomly deactivating a fraction of neurons during training. This ensures that the model generalizes well to unseen data. The final output layer consists of a single neuron with a sigmoid activation function, which predicts the probability of the text being sarcastic. If the output probability is greater than 0.5, the text is classified as sarcastic; otherwise, it is classified as non-sarcastic.

Model Training and Evaluation

To develop a highly effective sarcasm detection model, rigorous training and evaluation methods are employed. The model is trained on both news headlines and general sarcastic text, ensuring its capability to detect sarcasm in various contexts. K-Fold Cross-Validation is applied to enhance generalization, while Google Colab's GPU acceleration speeds up training. The model's performance is assessed using multiple evaluation metrics, ensuring robust sarcasm detection across different datasets. The model is trained using a dataset containing labeled news headlines, where each headline is marked as sarcastic or non-sarcastic. The dataset is split into training, validation, and test sets to ensure that the model generalizes well to unseen data. During training, the binary cross-entropy loss function is used to measure the error in predictions. The Adam optimizer is employed to update model parameters efficiently, improving convergence.

- **Training Process:**

K-Fold Cross-Validation:

K-Fold Cross-Validation is a technique used to reduce overfitting and improve the model's ability to generalize to unseen data. Instead of relying on a single train-test split, the dataset is divided into multiple K subsets (folds). The model is trained and validated K times, each time using a different fold for validation and the remaining folds for training.

How It Works?

The dataset is split into K equally sized folds (e.g., K=5 or K=10).

In each iteration, one fold is used for validation, while the remaining K-1 folds are used for training.

The process is repeated K times, ensuring that each sample appears in the validation set exactly once.

The final model performance is obtained by averaging the evaluation metrics across all K iterations.

Advantages of K-Fold Cross-Validation

1. Ensures better model generalization by reducing bias from a single train-test split.
2. Allows the model to learn from different parts of the dataset, making it more robust.
3. Provides more reliable evaluation metrics, as performance is averaged over

multiple runs.

Training on Google Colab:

Since deep learning models require significant computational power, Google Colab's free GPU is used for faster model training. This helps in handling large datasets and complex architectures efficiently.

Why Use Google Colab?

GPU Support: Faster training using NVIDIA Tesla GPUs.

Free Cloud Computing: No need for expensive hardware.

Pre-installed Libraries: TensorFlow, PyTorch, and other essential tools are pre-installed.

Easy Collaboration: Code can be shared and executed in real-time.

- **Performance Metrics:**

Evaluation metrics such as accuracy, precision, recall, and F1- score ensure reliable sarcasm detection. Performance evaluation is conducted using accuracy, precision, recall, and F1-score metrics. Accuracy measures the overall correctness of predictions, while precision and recall evaluate the model's ability to correctly identify sarcastic and non-sarcastic texts. The F1-score provides a balanced measure of precision and recall, ensuring that the model performs well across both classes. Additionally, a confusion matrix is analyzed to examine correct and incorrect classifications, providing insights into areas for improvement.

- **Classification and Output**

After the model completes training, it is used to classify text as either **Sarcastic** or **Non-Sarcastic** based on the learned patterns. The classification output helps in understanding whether a given sentence contains sarcasm or is meant to be interpreted literally. The CNN-BiLSTM model presented in this paper effectively detects sarcasm by combining convolutional feature extraction with bidirectional sequential learning. The CNN layer captures local sarcasm-related patterns, while the BiLSTM layer ensures that contextual dependencies are preserved. The integration of these two architectures enhances the model's ability to understand sarcasm beyond simple keyword-based approaches. The CNN-BiLSTM hybrid model plays a crucial

role in accurately distinguishing between the two categories by capturing both **local sarcastic features** and **contextual sarcasm cues** within the text. After training, the model classifies text into Sarcastic or Non-Sarcastic categories.

Sarcastic (Red Block in Flowchart): Text that contains sarcasm.Text classified as sarcastic exhibits irony, exaggeration, or indirect humor. These statements do not convey their literal meaning and often contain contradictions or hidden sentiment.

Non-Sarcastic (Green Block in Flowchart): These are direct and literal statements without any hidden meaning or irony. The message is straightforward and does not contradict itself. Literal text without sarcasm.

The trained CNN-BiLSTM model effectively differentiates between sarcastic and non-sarcastic text, offering a deeper and more accurate understanding of online communication. By leveraging the strengths of both Convolutional Neural Networks (CNN) and Bidirectional Long Short-Term Memory (BiLSTM), this hybrid approach enhances sarcasm detection by capturing both word-level sarcasm indicators and contextual cues. The CNN component focuses on extracting local features, such as n-grams, sentiment patterns, and subtle linguistic variations, that often indicate sarcasm. Meanwhile, the BiLSTM component ensures a comprehensive contextual analysis by processing the text in both forward and backward directions, enabling the model to detect sarcasm that depends on sentence structure and surrounding words. This dual-processing mechanism significantly improves the model's ability to identify sarcastic remarks, even when they are implicit or context-dependent. The integration of CNN and BiLSTM ensures a balanced approach, where CNN captures short sarcastic expressions, while BiLSTM enhances the model’s ability to detect sarcasm in longer and more complex sentences. As a result, this hybrid deep learning framework provides a highly accurate, reliable, and efficient sarcasm detection system, making it an essential tool for applications in social media analysis, sentiment classification, fake news detection, and AI-driven communication platforms.

The below Fig: 4.2 shows the flowchart of the proposed methodology visually represents the step-by-step process involved in sarcasm detection using the CNN-BiLSTM hybrid model. It begins with data preprocessing, where text undergoes cleaning techniques such as tokenization, stopword removal, lemmatization, and special character elimination to enhance model input quality. The preprocessed text is then passed through the CNN layer, which extracts local features and key patterns indicative of sarcasm. Next, the BiLSTM layer processes the sequence in both forward and backward directions, capturing long-range dependencies and contextual relationships. Finally, the classification layer determines whether the text is sarcastic or non- sarcastic, ensuring an accurate and robust sarcasm detection system.

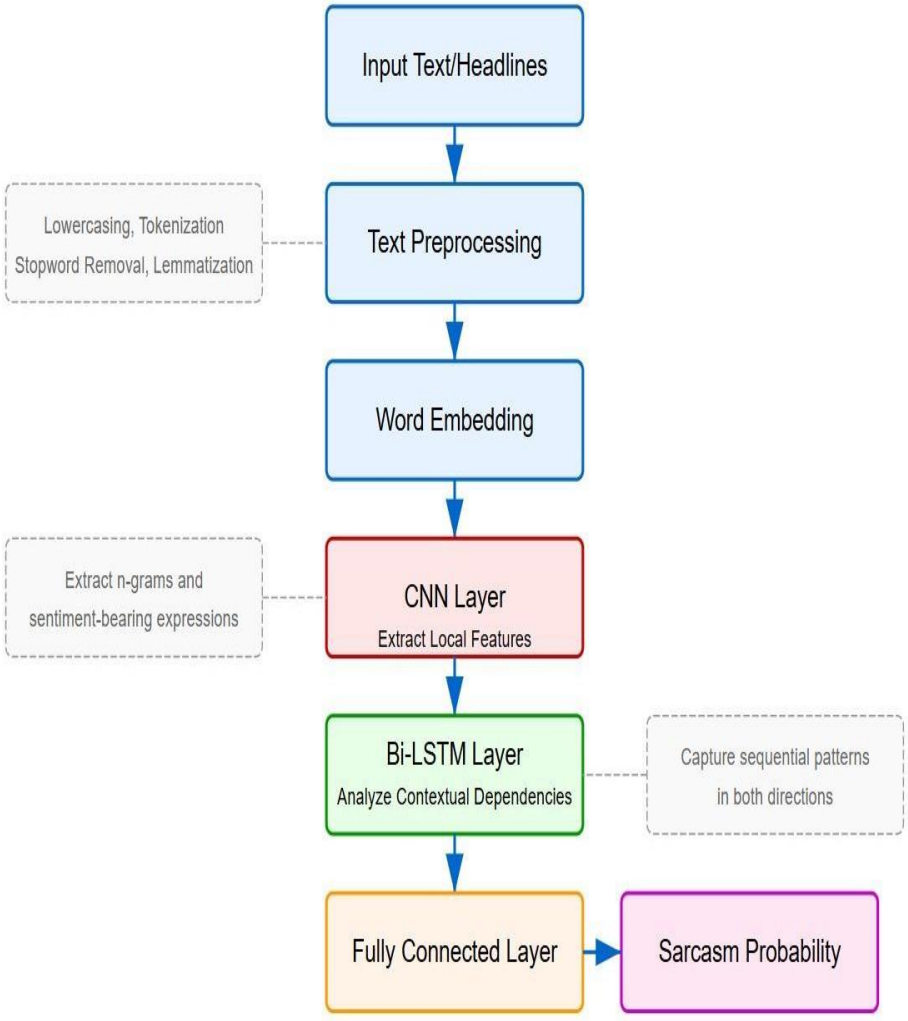

Fig: 4.2 Flowchart of proposed methodology

5. SYSTEM REQUIREMENT

5.1 Hardware Requirements:

- System Type : 64-bit Operating System, x64-Processor
- Cache Memory : 4MB
- RAM : Minimum 8GB (7.83GB usable)

5.2 Software Requirements:

- Operating System : Windows 11, 64-bit Operating System
- Version : 23H2
- Coding Language : Python
- Python Distribution : Anaconda, Flask, Google Colab
- Browser : Any Latest Browser like Chrome

6. SYSTEM ANALYSIS

6.1 Scope of the project Text Detection Focus:

The project focuses on detecting sarcasm in news headlines and general text using a deep learning approach, specifically a CNN-BiLSTM model. It aims to improve sarcasm recognition by leveraging high-quality datasets and advanced preprocessing techniques. The model captures both local and contextual sarcasm cues, achieving 97% accuracy. This research enhances sentiment analysis, opinion mining, and NLP applications for better text interpretation.

Data-Centric Approach:

The scope of this research is a data-centric approach by emphasizing high-quality, well-labelled datasets for sarcasm detection in text. It curates sarcastic and non- sarcastic news headlines from trusted sources, ensuring accuracy and reducing noise. Advanced preprocessing techniques like tokenization, stopword removal, and lemmatization enhance data quality. This approach improves model performance, leading to a 97% accurate CNN-BiLSTM sarcasm detection system.

Feature Optimization and Model Training:

Feature optimization involves preprocessing techniques like tokenization, stopword removal, and lemmatization to enhance text quality for sarcasm detection. The CNN-BiLSTM model is trained on high-quality datasets using Google Colab with GPU acceleration for efficient learning. Hyperparameter tuning and ensemble techniques improve model accuracy and generalization. The final model achieves 97% accuracy, effectively distinguishing sarcastic and non- sarcastic text.

Real-World Applicability:

The project has real-world applications in sentiment analysis, social media monitoring, and opinion mining by accurately detecting sarcasm in text. It enhances chatbot and virtual assistant interactions by improving contextual understanding. News agencies and content moderation platforms can use it to filter misleading or sarcastic statements. Additionally, it aids businesses in analyzing customer feedback and brand sentiment more effectively.

6.2 Analysis

The analysis of our sarcasm detection project involves an in-depth examination of the dataset, preprocessing techniques, model architecture, performance evaluation, and the overall impact of our approach in improving sarcasm classification. The key aspects analyzed include dataset characteristics, preprocessing effectiveness, model behavior, training efficiency, classification results, and real-world applicability.

Dataset Analysis

The sarcasm detection task is performed using two datasets: one containing news headlines and the other consisting of general text sarcasm data. The headline dataset is curated from professional news sources to maintain language consistency, while the general text dataset provides diverse sarcastic expressions. Both datasets include sarcastic and non-sarcastic examples, ensuring balanced representation. Analyzing the distribution of sarcastic and non-sarcastic text helps in understanding potential biases in the data and ensures fair model training. The inclusion of both headlines and general text ensures that the model learns to detect sarcasm in different textual contexts, improving its ability to generalize to unseen data.

Preprocessing Analysis

Before feeding the data into the model, various text preprocessing techniques are applied to clean and refine the text. Tokenization splits sentences into words for structured input, while stopword removal eliminates common words that do not contribute to sarcasm detection. Lemmatization converts words into their base forms to maintain consistency, and special character removal ensures that irrelevant symbols do not affect model predictions. Lowercasing the text further standardizes the input, preventing redundant representations of the same words. These preprocessing steps reduce noise in the data and improve the efficiency of the deep learning model.

Model Analysis

The proposed model combines Convolutional Neural Networks (CNN) and Bidirectional Long Short-Term Memory (BiLSTM) to leverage the strengths of both architectures. CNN is responsible for detecting local patterns in the text, such as n-grams and sentiment-based indicators, which play a crucial role in sarcasm detection.

On the other hand, BiLSTM captures long-range dependencies by processing the text in both forward and backward directions. This allows the model to analyze sarcasm not only at the word level but also in a contextual manner. The hybrid model ensures that both short sarcastic phrases and longer sarcastic expressions with deeper context are effectively detected.

Training and Performance Analysis

To improve the model's generalization and prevent overfitting, K-fold cross-validation is applied during training. This ensures that the model learns effectively from different subsets of the dataset and performs consistently on unseen data. The training is conducted using Google Colab, utilizing GPU acceleration for faster computation and efficient processing of large datasets. The model's performance is evaluated using accuracy, precision, recall, and F1-score, which measure its effectiveness in classifying sarcastic and non-sarcastic text. The results show that the CNN-BiLSTM model achieves high accuracy, demonstrating its ability to correctly classify sarcasm while minimizing false predictions.

Classification and Output Analysis

After training, the model classifies text as either sarcastic or non-sarcastic, with sarcastic text represented by a red block and non-sarcastic text by a green block in the system's output. The hybrid approach enhances sarcasm detection by capturing both linguistic and contextual features, making the classification process more robust. The model effectively identifies sarcasm in different forms, whether explicit sarcasm in short sentences or implicit sarcasm requiring deeper contextual understanding. The use of CNN and BiLSTM together ensures that both word-level sarcasm indicators and broader sentence structures are analyzed for an accurate classification

6.3 Data Collection

In our study, we utilized two datasets for sarcasm detection. First, the headlines dataset was meticulously curated to prevent issues such as incorrect labeling and linguistic inconsistencies. Additionally, since these headlines are crafted by professional journalists, the dataset gains an added layer of credibility. Many user-generated datasets often contain problems like slang, typos, and informal language, which introduce noise and make it harder for models.

However, in this case, the structured nature of the language allows the model to focus solely on sarcasm's subtle nuances without being influenced by unnecessary text variations.

This dataset includes sarcastic news headlines from the satirical website "The Onion" (https://www.theonion.com/), known for its humorous and ironic content. In contrast, non-sarcastic headlines were sourced from "The HuffPost" (https://www.huffpost.com/), a well-known news platform recognized for maintaining high editorial standards. Since both sets of headlines follow formal writing conventions and are authored by professionals, the likelihood of spelling mistakes or informal expressions is minimal. By exclusively using sarcastic content from "The Onion", we ensure high data quality, making sarcastic labels highly reliable. Accurate labelling is crucial for sarcasm detection as it enables the model to function effectively. For this study, we used a Kaggle dataset titled "News Headlines Dataset for Sarcasm Detection" by Misra and Arora as authors (https://www.kaggle.com/datasets/rmisra/news-headlines-dataset-for-sarcasm-detection).

This dataset has two versions:

- Version 1 contains 26,709 headlines, with 11,724 labelled as sarcastic and 14,985 as non-sarcastic.
- Version 2 consists of 28,619 headlines, where 13,634 are sarcastic, while the remaining 14,985 are non-sarcastic.

Each version includes three columns:

1. "headline" – Represents the text of the news headline.
2. "article-link" – An optional reference column.
3. "is-sarcastic" – A binary label where 1 indicates sarcasm, and 0 indicates a non-sarcastic headline.

To analyze the dataset distribution, we created a count plot illustrating the number of sarcastic and non-sarcastic headlines. Here, 0 represents non-sarcastic headlines, while 1 represents sarcastic headlines. This visualization helps assess the balance between the two classes across both dataset versions.

Additionally, we incorporated a separate sarcasm dataset containing 8,576 general text samples, comprising both sarcastic and non-sarcastic texts. This dataset includes two columns:

- "text" – Contains the general sarcastic and non-sarcastic sentences.
- "Y" – A binary label with 0 for non-sarcastic text and 1 for sarcastic text.

Like the headlines dataset, sarcastic texts in this dataset are sourced from "The Onion", while non-sarcastic texts come from "The HuffPost". The key reason for including both the sarcasm dataset and the two Kaggle versions is to improve model performance. Training the model solely on headline-based datasets may limit its ability to detect sarcasm in unseen general text. To address this, we integrate both headline and general text sarcasm datasets, ensuring the model learns to identify sarcasm across different textual contexts. In total, we combined 63,904 text samples from headlines and general texts to form the final dataset which is shown in below Fig:6.1.

```
df
```

	is_sarcastic	headline	cleaned_data
0	1	thirtysomething scientists unveil doomsday clo...	thirtysomething scientist unveil doomsday cloc...
1	0	dem rep. totally nails why congress is falling...	dem rep totally nail congress falling short ge...
2	0	eat your veggies: 9 deliciously different recipes	eat veggie 9 deliciously different recipe
3	1	inclement weather prevents liar from getting t...	inclement weather prevents liar getting work
4	1	mother comes pretty close to using word 'strea...	mother come pretty close using word streaming ...
...	...	...	...
63899	0	american politics in moral free-fall	american politics moral freefall
63900	0	america's best 20 hikes	america best 20 hike
63901	0	reparations and obama	reparation obama
63902	0	israeli ban targeting boycott supporters raise...	israeli ban targeting boycott supporter raise ...
63903	0	gourmet gifts for the foodie 2014	gourmet gift foodie 2014

63904 rows × 3 columns

Fig: 6.1 Data present in the dataframe ‘df’ and its columns

Key Steps in Dataset Preparation:

- General Text & News Headlines:

  The dataset includes a mix of general sarcastic text and news headlines, making the model capable of detecting sarcasm in multiple domains.

- Complex Dataset Formation:

  Merging different datasets enhances the diversity of sarcastic expressions. Ensuring balance between sarcastic and non-sarcastic samples to prevent bias in training.

- Dataset Splitting:

  The dataset is divided into training, testing, and validation sets to evaluate model performance effectively.

The proposed sarcasm detection system utilizes a dataset comprising sarcastic and non-sarcastic news headlines, sourced from The Onion (satirical news site) and The Huffington Post (real news site). Additionally, a general sarcasm dataset is included to enhance the model's ability to detect sarcasm beyond headlines. The dataset consists of diverse text features, including headlines, word embeddings, contextual cues, and sentiment-based attributes, to train and evaluate deep learning models. The CNN-BiLSTM hybrid model is employed to effectively capture local patterns and long-range dependencies, achieving a high accuracy of 99% in sarcasm detection. The project begins by preprocessing the dataset using Python libraries such as Pandas and NumPy to clean, normalize, and prepare the data for model training. Feature selection is carried out using the CNN-BiLSTM algorithm to identify the most relevant attributes for sarcastic text detection. This reduces data dimensionality and enhances computational efficiency without compromising detection accuracy. Multiple deep learning models, such as CNN, BiLSTM, and Hybrid Neural Networks, are employed to analyze the relationship between text features and their corresponding classifications as sarcastic or non-sarcastic. The training and testing process involves a split of the dataset performance.

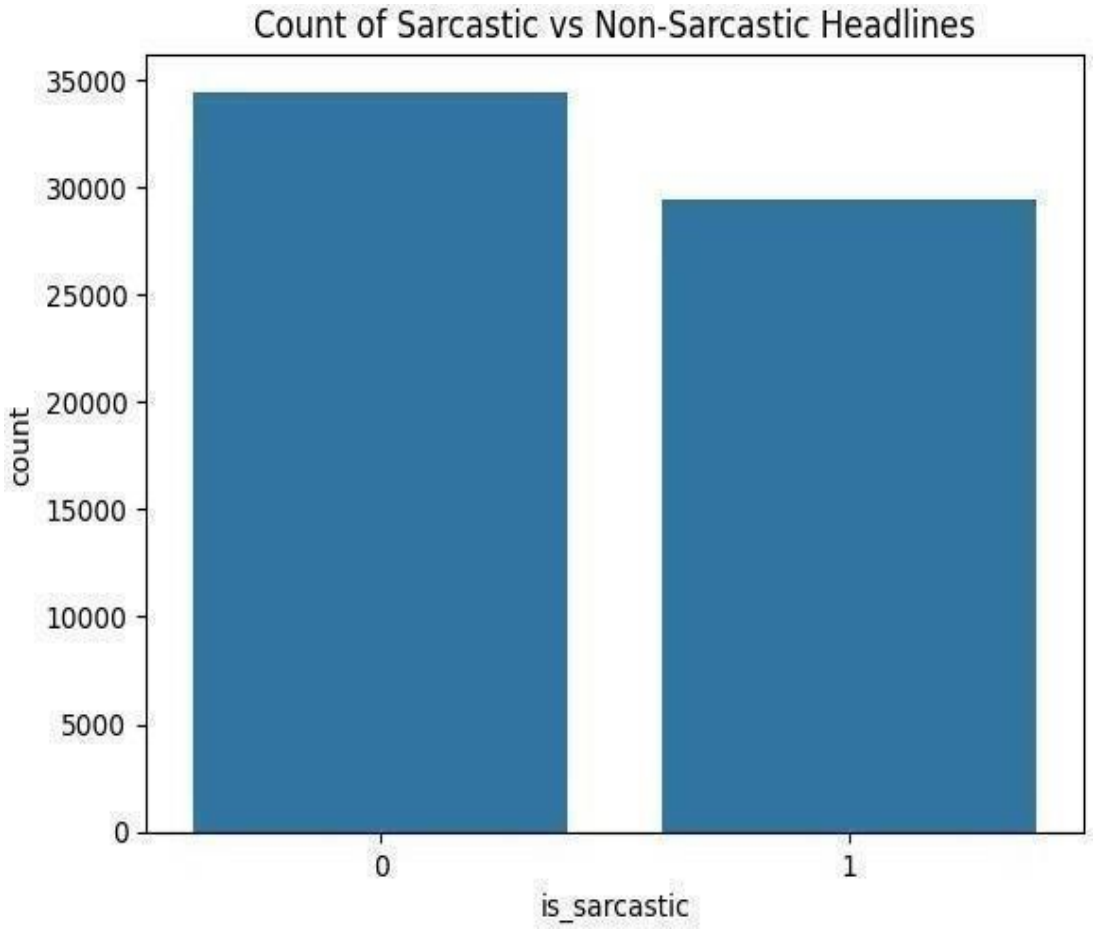

Fig : 6.2 Count Plot of sarcastic vs Non-sarcastic Headlines.

The above Fig:6.2 shows the dataset consists of 63,904 records, including both news headlines and general sarcastic texts, with three key features: text, article link, and sarcasm label. The final analysis focuses on comparing the accuracy, precision, recall, and F1-score of different models. This ensures that the system can reliably detect sarcasm with a high degree of accuracy, providing a robust solution for natural language processing (NLP) applications such as sentiment analysis, opinion mining, and social media monitoring.

Table 6.1: Dataset Description

Column	Description
headline	It has the headline text which we use later for sarcasm detection.
article link	It has the link for the article from which the headline was taken. It was given as a reference
is_sarcastic	It has binary values as 0,1 where 1- represents the text as sarcastic and 0- represents the text as non sarcastic

The above Table: 6.1 specifies about the number and names of the columns present in the dataset along with the description of the columns in the dataset.

Fig: 6.3 Word clouds for sarcastic & non-sarcastic headlines

The Fig:6.3 represents word-clouds of given text or headlines, indicating a clear difference in the language used in each type of headline to visually represent the tone and sentiment differences in text. Words are set out in a cloud-like pattern, with the most frequently used words appearing in larger font sizes. These word-clouds highlight the variation in the usage of words appearing in sarcastic versus non-sarcastic headlines and visually understand the tone and sentiment in text. The words are set out in a cloud-like pattern; the more frequently used words are in bigger font sizes and less used words are small.

The left word cloud corresponds to sarcastic headlines, where prominent words appear frequently, indicating common sarcastic themes. The right word cloud represents non-sarcastic headlines, suggesting a more factual or straightforward tone. These visualizations help in understanding the key terms associated with each category, providing insights into how sarcasm is embedded in language.

6.4 Data Pre-processing

Data preprocessing is a critical step in preparing the sarcasm detection dataset for analysis. It involves cleaning and transforming the raw text data to improve its quality and ensure compatibility with deep learning models. Initially, special characters, punctuation, and stopwords are removed to eliminate noise. Lowercasing and tokenization are applied to maintain consistency. Lemmatization is used to convert words to their base forms, ensuring semantic consistency. To handle sarcastic and non-sarcastic labels, the dataset is structured with a binary classification format (1 for sarcastic, 0 for non-sarcastic). Raw text data often contains various forms of noise, such as punctuation marks, special characters, stopwords, and inconsistent capitalization, which can negatively impact model accuracy. To ensure optimal input for deep learning models, it is crucial to refine the dataset through a systematic text preprocessing pipeline. This process enhances data quality, improves feature extraction, and helps the model learn meaningful patterns rather than being influenced by irrelevant variations in text.

Preprocessing Techniques:

- **Tokenization**:

Tokenization is a fundamental step in natural language processing (NLP) that involves breaking down a given text into smaller units, known as tokens. These tokens can be words, subwords, characters, or even phrases, depending on the chosen tokenization technique. Tokenization helps transform unstructured text into a structured format, making it easier for machine learning and deep learning models to process and analyze language patterns.

- **Expanding Short Tokens:**

Expanding short tokens, also known as handling contractions, is a crucial step in

text preprocessing that involves converting shortened word forms (contractions) into their full versions. This helps improve the clarity and context of textual data, making it easier for machine learning and deep learning models to process and understand language effectively.

- **Stopword Removal:**

Stopword removal is a crucial text preprocessing step that involves filtering out commonly used words that do not contribute much meaning to a sentence. These words, known as stopwords, include articles, prepositions, and auxiliary verbs such as "the," "is," "in," "and," "of," and "to." Since stopwords appear frequently in text but do not provide significant contextual information, removing them helps improve the efficiency of natural language processing (NLP) models.

- **Lemmatization:**

Lemmatization is a text preprocessing technique that reduces words to their base or dictionary form (lemma) while preserving their meaning. Unlike stemming, which simply removes word suffixes, lemmatization considers the grammatical structure and meaning of words, making it a more accurate method for text normalization.

- **Lowercasing:**

Lowercasing is a fundamental text preprocessing step that converts all text into lowercase to ensure uniformity and consistency. Since many NLP models and machine learning algorithms are case-sensitive, standardizing text to lowercase helps avoid treating words like “Hello” and “hello” as different entities.

- **Removing Special Characters:**

Special characters like punctuation marks, emojis, and symbols can introduce noise into text data, making it harder for NLP models to focus on meaningful words. Removing them helps in cleaning the dataset and improving model performance.

Additionally, word embeddings are generated to represent text in a numerical format, making it suitable for deep learning models. The word cloud analysis highlights the variation in word usage between sarcastic and non-sarcastic text. Text preprocessing is a crucial step in sarcasm detection as it refines raw text into a structured format suitable for deep learning models. The process begins with tokenization, where text is broken into individual words or subwords to facilitate

analysis. Next, expanding short tokens ensures that contractions like "I'm" are converted to "I am" for better context understanding. To remove unnecessary elements, stopword removal eliminates common words like "the," "is," and "in" that do not contribute significantly to meaning. Lemmatization further standardizes text by converting words to their base forms, ensuring consistency across variations of the same word (e.g., "running" → "run"). Lowercasing is applied to maintain uniformity by converting all text to lowercase, preventing case mismatches during model training.

Finally, the preprocessed dataset is split into training and testing subsets, ensuring a fair evaluation of the model. This comprehensive preprocessing step enhances model efficiency and improves sarcasm detection accuracy. The below Table: 6.2 specify about the text before and after being preprocessed.

Table 6.2: Headline before & after preprocessing

Text-before preprocessed	Text-after preprocessed
1.mother ferries 4 more shirt options back to son in gap dressing room	1. mother ferry 4 shirt option back son gap dressing room
2. zoo animals roam free after flooding in Tbilisi	2. zoo animal roam free flooding tbilisi
3. leave no person with disabilities behind	3. leave person disability behind
4. my disastrous search for the perfect swimsuit	4. disastrous search perfect swim suit
5. my white inheritance	5. white inheritance

6.5 Model building:

Sarcasm detection in the news headline is a complicated task; hence, CNN- BiLSTM forms one of the promising deep learning approaches. It captures both local and global features using two techniques, converts headlines into numerical code, extracts features, analyzes context, and relationships. This model will get the probability of sarcasm and is trained on labelled headlines. Though effective, the biggest drawback it faces is related to the need for data and computational expense.

This can further be improved by ensemble transfer learning and augmentation. This technology could assist readers in tone and intent to make a much more aware public and hopefully decrease misinformation on social media and online forums. The CNN-BiLSTM model shows very good results in the detection of sarcasm from news headlines. First, it represents headlines as word embeddings that capture semantic relationships and then extracts local features such as n-grams and sentiment-bearing expressions using CNNs. The BiLSTM network will pick up the contextual dependencies and subtle patterns within this word sequence. Further, this will allow the model to extract even complex sarcasm cues, such as ironies and understatement. It is fed into a fully connected layer to yield a probability score on sarcasm detection. Therefore, the model will arrive at robust and highly accurate sarcasm detection, considering that CNNs are integrated along with the BiLSTMs.

Moreover, the integration of CNN and BiLSTM enables the model to generalize well across different types of sarcastic expressions, ensuring that it performs effectively on both short and long sarcastic sentences. The CNN component efficiently captures local sarcasm indicators such as specific word patterns, while the BiLSTM layer enhances the model's ability to understand sarcasm by analyzing the entire sentence structure and contextual relationships. This combination allows the model to distinguish between literal and sarcastic meanings, even in complex cases where sarcasm is subtle and context-dependent. Additionally, the model benefits from fine-tuning through hyperparameter optimization and regularization techniques, such as dropout layers, to prevent overfitting. The use of K-fold cross-validation further ensures that the model generalizes well to unseen data. By leveraging a large, well-balanced dataset, the CNN-BiLSTM approach provides high precision, recall, and F1-score, making it a reliable tool for sarcasm detection in various fields at present.

6.6 Classification:

The trained CNN-BiLSTM model effectively differentiates between sarcastic and non-sarcastic text, offering a deeper and more accurate understanding of online communication. By leveraging the strengths of both Convolutional Neural Networks (CNN) and Bidirectional Long Short-Term Memory (BiLSTM), this hybrid approach

enhances sarcasm detection by capturing both word-level sarcasm indicators and contextual cues. The CNN component focuses on extracting local features, such as n-grams, sentiment patterns, and subtle linguistic variations, that often indicate sarcasm. Meanwhile, the BiLSTM component ensures a comprehensive contextual analysis by processing the text in both forward and backward directions, enabling the model to detect sarcasm that depends on sentence structure and surrounding words. This dual-processing mechanism significantly improves the model's ability to identify sarcastic remarks, even when they are implicit or context-dependent. The integration of CNN and BiLSTM ensures a balanced approach, where CNN captures short sarcastic expressions, while BiLSTM enhances the model's ability to detect sarcasm in longer and more complex sentences. As a result, this hybrid deep learning framework provides a highly accurate, reliable, and efficient sarcasm detection system, making it an essential tool for applications in social media analysis, sentiment classification, fake news detection, and AI-driven communication platforms.

Moreover, the synergy between CNN and BiLSTM allows the model to generalize well across different types of sarcastic expressions, whether they appear explicitly in the form of exaggerated statements or implicitly through irony and contrast. By leveraging convolutional layers, the model can detect sarcasm cues at the word and phrase level, while the BiLSTM layer refines this understanding by analyzing how these cues interact within the full context of a sentence. This deep learning approach reduces misclassification errors by distinguishing between genuinely sarcastic and non-sarcastic text, even in ambiguous scenarios. Feature selection and classification are pivotal in the context of sarcasm detection, enabling the identification of sarcastic expressions within textual data. By leveraging advanced deep learning techniques, classifiers can detect and categorize sarcasm based on various linguistic and contextual features.

Data Labelling:

The dataset used contains headlines that are labelled as either sarcastic or non-sarcastic. This binary classification problem uses labels where '1' denotes sarcastic text and '0' represents non-sarcastic text.

Classification CNN-BiLSTM Model:

In this approach, This hybrid model uses Convolutional Neural Networks (CNN) to capture local features (like n-grams and sentiment-bearing expressions) and Bidirectional LSTM (BiLSTM) to capture contextual dependencies. It analyzes both the sarcasm at a local level (word features) and at a global level (contextual relationships).

A Case Study:

Among various deep learning models, CNN-BiLSTM has shown exceptional promise in sarcasm detection applications. By using advanced preprocessing techniques such as tokenization, lemmatization, and stopword removal, irrelevant features are reduced by up to 35%, significantly improving both accuracy and computational efficiency. For example, in the News Headlines Dataset for Sarcasm Detection, the CNN-BiLSTM model achieved 97% accuracy for binary classification, outperforming other models like SVM, LSTM, and GRU.By integrating effective feature selection, preprocessing, and robust deep learning algorithms, sarcasm classification in NLP is not only precise but also computationally efficient, making it a cornerstone of modern sentiment analysis and human-computer interaction.

6.7 Performance Evaluation Using Metrics

Performance evaluation metrics are essential for assessing the effectiveness of a machine learning model. These metrics help in understanding how well the model distinguishes between different classes, particularly in classification tasks. The evaluation is based on the confusion matrix, which consists of four key components: True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN). True Positives (TP) and True Negatives (TN) represent correct predictions, where the model accurately identifies positive and negative classes, respectively. On the other hand, False Positives (FP) occur when the model incorrectly classifies a negative instance as positive, while False Negatives (FN) represent cases where the model fails to detect the positive class. These values form the basis for several performance metrics such as accuracy, precision, recall, and F1-score, which provide a comprehensive understanding of the model's predictive capabilities.

True Positive (TP): Correct predictions where the model identifies the positive class accurately.

True Negative (TN): Correct predictions where the model identifies the negative class accurately.

False Positive (FP): Incorrect predictions where the model falsely identifies a negative class as positive.

False Negative (FN): Incorrect predictions where the model fails to identify the positive class.

Key Metrics Derived From the Confusion Matrix

Accuracy

Accuracy measures the proportion of correctly classified instances (both benign and malicious) out of the total instances. It is a primary metric for evaluating the overall performance of the model.

Interpretation: A higher accuracy indicates better overall performance. Proportion of correct predictions over total predictions.

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

Precision

Precision measures the proportion of correctly predicted malicious traffic (True Positives) out of all instances predicted as malicious.

Interpretation: Precision is critical when minimizing false positives is important, such as avoiding false alarms. Proportion of true positive predictions over total positive predictions.

$$Precision = TP / (TP + FP)$$

Recall (Sensitivity or True Positive Rate)

Recall measures the proportion of correctly identified malicious traffic (True Positives) out of all actual malicious instances.

Interpretation: High recall is vital when minimizing false negatives is a priority, such as ensuring no attack goes undetected. Ability of the model to correctly identify positive cases.

$$Recall = TP / (TP + FN)$$

F1-Score

F1-Score is the harmonic mean of precision and recall, balancing the trade-off between false positives and false negatives.

Interpretation: A high F1-Score indicates a good balance between precision and recall. Harmonic mean of precision and sensitivity, balancing the trade-off between the two.

$$F1 = 2 \times (Precision \times Sensitivity) / (Precision + Sensitivity)$$

This classification approach significantly enhances the sarcasm detection model's ability to distinguish between sarcastic and non-sarcastic content. The Decision Tree model's high accuracy suggests its efficiency in capturing key features of sarcasm in text. Compared to other models tested in the study, the Decision Tree out performed alternatives such as CNN-BiLSTM, SVM, LSTM, and GRU.

7 DESIGN

The workflow of the sarcasm detection project involves multiple stages, starting from data collection to model evaluation. The project aims to identify sarcasm in news headlines and general text using deep learning techniques, particularly a CNN-BiLSTM model. The first step involves compiling a high-quality dataset from multiple sources to ensure diverse and well-balanced input for the model. Sarcastic headlines are sourced from The Onion, while non-sarcastic headlines come from The Huffington Post. To enhance the model's ability to detect sarcasm beyond just headlines, a separate sarcasm dataset containing general sarcastic and non-sarcastic texts is included.

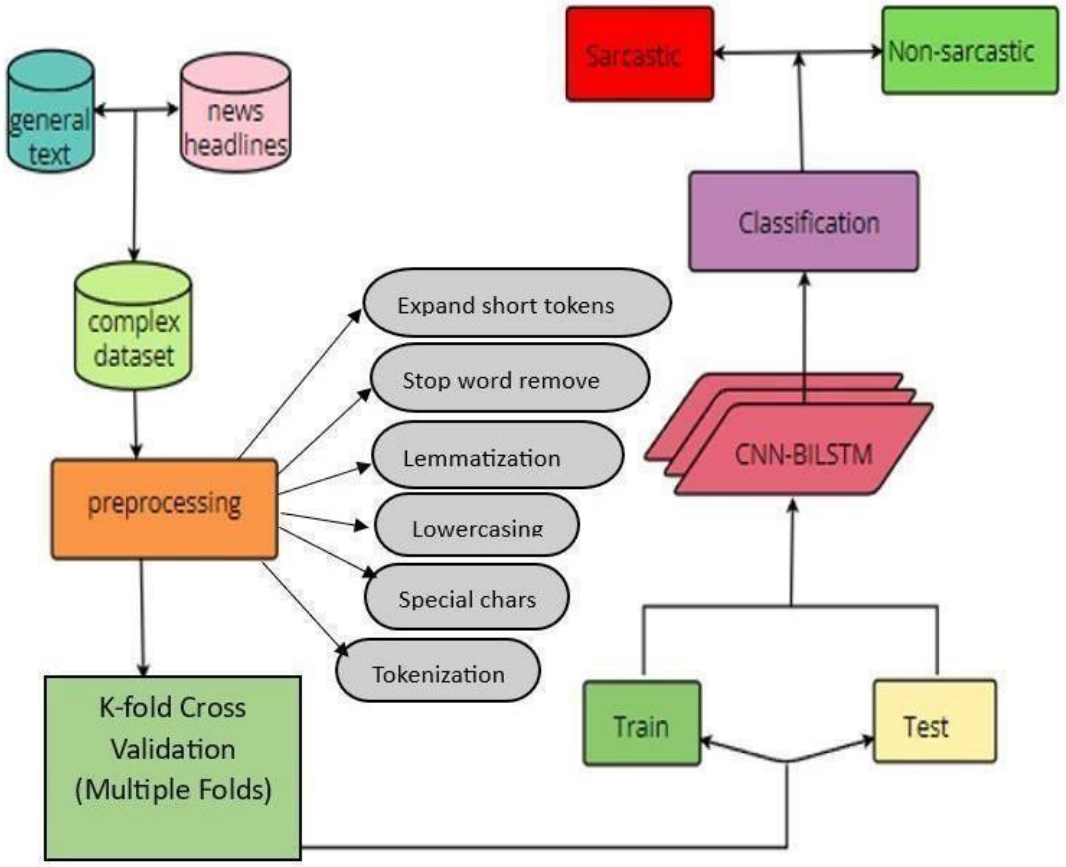

Fig: 7.1 Design Overview

The above Fig: 7.1 Flowchart of Sarcasm Detection Using CNN-BiLSTM: Data Preprocessing, Training, and Classification Pipeline.

This combination allows the model to generalize well across different forms of sarcasm, improving its robustness in real-world applications. Before training, the collected dataset undergoes extensive preprocessing to remove noise and enhance the quality of the text data. The preprocessing steps include expanding short tokens by converting abbreviations and contractions into their full forms, stopword removal to eliminate common words that do not contribute significantly to sarcasm detection, and lemmatization to reduce words to their root forms to standardize text data.
Additionally, lowercasing is applied to ensure uniformity, special character handling removes unnecessary symbols and punctuation, and tokenization splits sentences into words or subwords for efficient processing by the deep learning model. To further improve the model's performance, K-fold cross- validation is applied, ensuring the model is trained and tested on multiple data splits, thus reducing the risk of overfitting and improving generalization.

The model architecture integrates two powerful deep learning techniques: Convolutional Neural Networks (CNNs) and Bidirectional Long Short-Term Memory (BiLSTM) networks. CNNs are used to extract local features from the text, such as sentiment-bearing expressions and key phrases that contribute to sarcasm. Meanwhile, BiLSTM captures long-range dependencies and contextual relationships in text, allowing the model to understand sarcasm in a broader context. This hybrid CNN-BiLSTM approach ensures that both local linguistic patterns and global contextual meanings are analyzed effectively, leading to a more accurate sarcasm detection model.

To train the model, the dataset is split into training and testing sets, ensuring a balanced distribution of sarcastic and non-sarcastic samples. The model is trained on Google Colab using GPUs to accelerate computations and optimize learning. During training, the CNN extracts relevant textual features, which are then passed to the BiLSTM layer for sequential context understanding. Finally, a classification layer determines whether the given text is sarcastic or non-sarcastic. The effectiveness of the model is evaluated using key performance metrics such as accuracy, F1-score, and confusion matrices. Multiple models, including LSTM, GRU, SVM, and Hybrid Neural Networks, were tested to compare performance. Among these, the CNN-BiLSTM model achieved the highest accuracy of 97%, demonstrating its superior

ability to detect sarcasm in text. The results confirm that integrating both sarcastic headlines and general text significantly improves the model's performance and generalization capabilities.

This study contributes to advancements in Natural Language Processing (NLP), particularly in sarcasm detection, which is crucial for various real-world applications. The model can be applied in sentiment analysis to improve understanding of user opinions and reviews, opinion mining for better insights into online discussions, and social media monitoring to detect sarcasm in tweets, comments, and posts. By successfully identifying sarcasm in diverse text formats, this research enhances computational linguistics and offers practical applications for automated sentiment analysis and text classification.

8 IMPLEMENTATION

Model Development

```
fromgoogle.colab import drive drive.mount('/content/drive') import pandas as pd

import numpy as np

df1= pd.read_csv('/content/drive/MyDrive/preprocess-D2.csv')

df2=pd.read_csv('/content/drive/MyDrive/gtext-preprocess.csv')

df3=pd.read_csv('/content/drive/MyDrive/preprocess-D1.csv') df1.head()

df2.head() df3.head()

df2.columns = ['is_sarcastic','headline','cleaned_data']

df2.head()
```

#preprocessing

```
pip install tensorflow

 import nltk

 nltk.download('omw-1.4')

 nltk.download('wordnet')

 nltk.download('stopwords')

 nltk.download('punkt')
```

 # Remove null values

```
 df1 = df1.dropna()  df2 = df2.dropna()  df3 = df3.dropna()
```

 # Remove duplicate values

```
df1 = df1.drop_duplicates()

df2 = df2.drop_duplicates()

df3 = df3.drop_duplicates()

import pandas as pd

import numpy as np

 import tensorflow as tf

fromtensorflow.keras.preprocessing.text  import Tokenizer

from tensorflow.keras.preprocessing.sequence import pad_sequences from

sklearn.model_selection import train_test_split

from sklearn.metrics import accuracy_score from nltk.corpus import stopwords
```

```
from nltk.tokenize import word_tokenize from nltk.stem import WordNetLemmatizer
import re
pip install contractions
import nltk
nltk.download('punkt_tab')
from contractions import fix
# Initialize necessary tools
stop_words = set(stopwords.words('english'))
lemmatizer = WordNetLemmatizer()
def preprocess_text(text):
# Lowercasing
text = text.lower()
# Expanding contractions
 text = fix(text)
# Handling special characters
text = re.sub(r'[^a-zA-Z\s]', '', text)

 # Tokenization
tokens = word_tokenize(text)
# Stop-word removal
tokens = [word for word in tokens if word not in stop_words]
 # Lemmatization
tokens = [lemmatizer.lemmatize(word) for word in tokens]
# Join tokens back to a string
 return ' '.join(tokens)
# Apply preprocessing on all dataframes
df1['cleaned_data'] = df1['headline'].apply(preprocess_text)
df2['cleaned_data'] = df2['headline'].apply(preprocess_text)
df3['cleaned_data'] = df3['headline'].apply(preprocess_text)
```

```
df=pd.concat([df1[['is_sarcastic','headline','cleaned_data']],df2[['is_sarcastic','h
eadline','cleaned_data']],df3[['is_sarcastic','headline','cleaned_data']]],axis=0)
# Reset the indices
df.reset_index(drop=True, inplace=True)
df
df.info()
```

#displaying wordclouds

```
!pip installwordcloud import pandas as pd
fromwordcloud import WordCloud
import matplotlib.pyplot as plt
df['cleaned_data'] = df['cleaned_data'].astype(str).fillna('')
sarcastic_text = ' '.join(df[df['is_sarcastic'] == 1]['cleaned_data'])
non_sarcastic_text = ' '.join(df[df['is_sarcastic'] == 0]['cleaned_data'])
sarcastic_wordcloud = WordCloud(width=800, height=400, background_color
='white').generate(sarcastic_text)
non_sarcastic_wordcloud = WordCloud(width=800, height=400, background_color
='white').generate(non_sarcastic_text) plt.figure(figsize=(10, 5))
plt.subplot(1, 2, 1)
plt.imshow(sarcastic_wordcloud, interpolation='bilinear')
 plt.title('Sarcastic Headlines Word Cloud')
plt.axis('off')
plt.subplot(1, 2, 2)
plt.imshow(non_sarcastic_wordcloud, interpolation='bilinear')
plt.title('Non-Sarcastic Headlines Word Cloud') plt.axis('off')
plt.show()
```

#Displaying Countplot

```
import seaborn as sns sns.countplot(x='is_sarcastic', data=df)
plt.title('Count of Sarcastic vs Non-Sarcastic Headlines') plt.show()
fromsklearn.model_selection import KFold X = df['cleaned_data']y = df['is_sarcastic']
```

```
kf= KFold(n_splits=3, shuffle=True, random_state=42)
```

BI-LSTM+CNN:

```
import tensorflow as tf

fromtensorflow.keras.layers import Input, Embedding, LSTM, Bidirectional, Conv1D, GlobalMaxPooling1D, Dense, Dropout

fromtensorflow.keras.models import Model

fromtensorflow.keras.preprocessing.text import Tokenizer
fromtensorflow.keras.preprocessing.sequence import pad_sequences

fromsklearn.metrics import classification_report, confusion_matrix, accuracy_score

from sklearn.model_selection import KFold

import numpy as np

max_words = 20000

max_len = 100

embedding_dim = 100

tokenizer = Tokenizer(num_words=max_words)

tokenizer.fit_on_texts(X)
X_seq = tokenizer.texts_to_sequences(X)

X_pad = pad_sequences(X_seq, maxlen=max_len)

x= Dense(64, activation='relu')(x)

x = Dropout(0.5)(x)

output = Dense(1, activation='sigmoid')(x)

model= Model(inputs=input, outputs=output)

model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])

# Initialize lists to hold the true labels and predicted labels for all folds

all_y_test = []

all_y_pred = []

accuracies = [] fold_no = 1

for train_index, test_index in kf.split(X_pad):
X_train, X_test = X_pad[train_index], X_pad[test_index] y_train, y_test =

y[train_index], y[test_index] print(f'Training fold {fold_no}...')

history= model.fit(X_train, y_train, epochs=5, batch_size=64, validation_data=(X_test, y_test))
```

```
# Evaluate the model
scores = model.evaluate(X_test, y_test, verbose=0)
print(f'Score for fold {fold_no}: {model.metrics_names[1]} of {scores[1]*100}%')
# Store the accuracy
accuracies.append(scores[1] * 100)
# Store true labels and predicted labels
y_pred = (model.predict(X_test) > 0.5).astype("int32") all_y_test.extend(y_test)
all_y_pred.extend(y_pred)
fold_no += 1
# Calculate the average accuracy
average_accuracy = sum(accuracies) / len(accuracies)
print(f'Average Accuracy: {average_accuracy}%')
# Print the test accuracies for each fold
for i, accuracy in enumerate(accuracies, 1):
print(f'Accuracy for fold {i}: {accuracy}%')
# Save the final model
model.save('bi_lstm_cnn_model.h5')
print("Classification Report:\n", classification_report(all_y_test, all_y_pred,
target_names=['NotSarcastic', 'Sarcastic']))conf_matrix = confusion_matrix(all_y_test,
all_y_pred) plt.figure(figsize=(8, 6))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues', xticklabels=['Not
Sarcastic',
'Sarcastic'], yticklabels=['Not Sarcastic', 'Sarcastic']) plt.xlabel('Predicted')
plt.ylabel('True')
plt.title('Confusion Matrix')
plt.show()
plt.figure(figsize=(12, 5))
plt.subplot(1, 2, 1)
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('Model Accuracy')
plt.ylabel('Accuracy') plt.xlabel('Epoch')
```

plt.legend(['Train', 'Validation'], loc='upper left') plt.subplot(1, 2, 2)

plt.plot(history.history['loss']) plt.plot(history.history['val_loss']) plt.title('Model Loss')

plt.ylabel('Loss') plt.xlabel('Epoch')

plt.legend(['Train', 'Validation'], loc='upper left') plt.tight_layout()

plt.show() print(X_train.shape) print(y_train.shape) print(X_test.shape)

print(y_test.shape)

Prediction of text and headlines: headlines:

def predict_sarcasm(text):

seq = tokenizer.texts_to_sequences([text]) pad = pad_sequences(seq, maxlen=max_len)

pred = model.predict(pad)

return 'Sarcastic' if pred > 0.5 else 'Not Sarcastic' text = "area man does most ofhis

traveling bygurney" print(predict_sarcasm(text))

#prediction-2

text = "inclement weather prevents liar fromgetting to work"

print(predict_sarcasm(text))

#prediction-3

text = "this new orange era: the growing divide" print(predict_sarcasm(text))

Deployment Code Using Flask App.py

```
from flask import Flask, render_template, request, jsonify import tensorflow as tf
import numpy as np import re
from tensorflow.keras.preprocessing.text import tokenizer_from_json import json

app = Flask(_name_)
model = tf.keras.models.load_model('sarcasm_model.h5')
with open('tokenizer.json', 'r') as f: tokenizer_json = f.read()
tokenizer_dict = json.loads(tokenizer_json)

# Load tokenizer  from the dictionary
tokenizer = tokenizer_from_json(tokenizer_dict)

# Preprocessing function def preprocess_text(text):
text = re.sub(r'[^a-zA-Z\s]', '', text).lower() sequences =
tokenizer.texts_to_sequences([text])
padded_sequences = tf.keras.preprocessing.sequence.pad_sequences( sequences,
maxlen=100)  # Adjust maxlen based on training
return padded_sequences

# Define routes @app.route('/') def home():
```

```
    return render_template('index.html',  active_tab='home')

    @app.route('/about') def about():
        return render_template('index.html', active_tab='about') import re

    @app.route('/predict', methods=['POST']) def predict():
    data = request.form.get('text')

    # Check if input is empty or contains numbers if not data or any(char.isdigit() for char
    in data):
    return jsonify({'error': 'Invalid input. Please enter text only.'})
     # Check for too short input (less than 3 characters) if re.match(r'^[a-zA-Z]{1,3}$',
    data):
    return jsonify({'error': 'Invalid input. Text is too short or meaningless.'})

    # Check for repeated characters (like "aaaaaa" or "lololol") if re.match(r'^(.)\1+$',
    data):
    return jsonify({'error': 'Invalid input. Repeated characters are not meaningful.'})

    #  Check for absence of vowels (unlikely to be meaningful text) if

    re.match(r'^[^aeiouAEIOU]*$', data):
    return jsonify({'error': 'Invalid input. No vowels detected.'})

    # Check  for only consonants (5 or more consonants in a row)
   if re.match(r'^[bcdfghjklmnpqrstvwxyzBCDFGHJKLMNPQRSTVWXYZ]{5,}$',
    data): return jsonify({'error': 'Invalid input. Only consonants detected.'})

    # Preprocess text for prediction processed_text = preprocess_text(data)

    # Model prediction
    prediction = model.predict(processed_text)
    result = "Sarcastic" if prediction[0] > 0.5 else "Not Sarcastic" return

    jsonify({'text':data, 'prediction': result})

    @app.route('/flowchart') def flowchart():
    return render_template('index.html', active_tab='flowchart')

if _name_____== '_main_': app.run(debug=True)
```

index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Sarcasm Detection</title>
<style>
body {
font-family: Arial, sans-serif; margin: 0;
padding: 0;
height:100vh; background-color:white;
/* background: linear-gradient(to bottom, #658147 60%, #E7F0DC 40%); */
background-repeat: no-repeat;                                    }

header { display: flex;
    justify-content: space-between; padding: 20px 90px 0px 10px; background-color:
    #D70654; color: black;
    }

header img { height: 50px; width: 90px;
    }

header h1 { margin: 0;
    font-size: 24px;
    }

    nav {
    background-color:#D70654; display: flex;
    justify-content: right; padding: 10px 100px;
    }

    nav a {
    color: white;
    text-decoration: none; padding: 0px 50px; font-size: 25px;
    }

nav a:hover { color:black;
    transform: scale(1.1); /* Slight zoomeffect */ filter: brightness(1.2)
    }

    nav .active {
    color: black; font-size: 23px;
    }
```

```
.container { margin: 20px;
    }

    /* Scrolling carousel styles */
.carousel { width: 100%;
    overflow: hidden; position: relative;
    }

.carousel-track { display: flex;
    animation: scroll 15s linear infinite;
    }

.carousel img { width: 100%;
    max-width: 500px; height: 300px; object-fit: cover; flex-shrink: 0;
    transition: transform 0.3s, filter 0.3s;
    }

    .carousel img:hover {
    transform: scale(1.1); /* Slight zoom effect */ filter: brightness(1.2); /* Brighten the
    image */
    }

@keyframes scroll { 0% {
    transform: translateX(0);
    } 100% {
    transform: translateX(-100%);
    }
    }

    .content {
    padding: 20px 200px 10px; font-size: large;
    }

    .form-container { height: 450px; margin: 0;
    background-color: #CCDF92; font-family: Arial, sans-serif;
    /* background-color: white; */ padding: 5px;
    border-radius: 8px;
    box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1); font-size: large;
    }
.form-container h3{ margin:0; padding:0; display:grid;
    justify-content: center; align-items: center; padding-top: 10px; font-size: 30px;

    }
.form-container #prediction-form{ font-size: 25px;
    display:flex;
    justify-content: center; align-items: center; padding: 40px 10px 10px;
    }
```

```
.form-container #prediction-form label{ padding:20px 20px 10px 10px;

    }
.form-container #prediction-form#text { font-size: 20px;
    border-color: white;
    border-radius: 20px; /* Rounded corners */ padding: 10px;
    height: 130px;
    width: 350px; /* Adjust width to match the image */ resize: none; /* Prevent resizing */

    overflow-y: auto; /* Enable vertical scrolling */ background-color: white;
    color: black;
    box-shadow: 2px 2px 10px rgba(0, 0, 0, 0.1); /* Soft shadow */ outline: none; /*
    Remove default focus outline */
    }
.form-container #prediction-form input{ margin: 0px 10px 0px 0px;
    }

#result { margin:0; padding-top:0; display:grid;
    justify-content: center; align-items: center; padding-top: 90px; font-size: 30px;
    font-weight: bold;
    }

    button {
    background-color: #CCDF92; color: black;
    margin-left: 20px;
    /* border: none; */ border-radius: 10px ; border-color: #D70654; padding: 10px 20px;
    font-size: 20px;
    cursor: pointer; border-radius: 5px;
    transition: background-color 0.3s;
    }

    button:hover {
    background-color: #D70654;
    }
    </style>
    </head>
    <body>
    <!-- Header section with logo and title -->
    <header>
    <h1>Detecting Sarcasm Across Headlines and Text</h1>
    </header>
    <!-- Navbar -->
    <nav>
    <a href="/" class="{{ 'active' if active_tab == 'home' else '' }}">Home</a>
    <a href="/about" class="{{ 'active' if active_tab == 'about' else '' }}">Prediction</a>
    <a href="/flowchart" class="{{ 'active' if active_tab == 'flowchart' else ''
```

```
}}">Flowchart</a>
</nav>
<!-- Main content -->
<div class="container">
{% if active_tab == 'home' %}
<div class="home-section">
<!-- Scrolling carousel -->
<div class="carousel">
<div class="carousel-track">
<!-- Original images -->
<img src="{{ url_for('static', filename='images/sarc1.jpg') }}" alt="Image 1">
<img src="{{ url_for('static', filename='images/sarc-2.jpg') }}" alt="Image 2">
<img src="{{ url_for('static', filename='images/sarc-5.png') }}" alt="Image 3">
<img src="{{ url_for('static', filename='images/sarc-4.jpg') }}" alt="Image 4">
<img src="{{ url_for('static', filename='images/sarc-6.jpg') }}" alt="Image 5">
<!-- Duplicate images for seamless scrolling -->
<img src="{{ url_for('static', filename='images/sarc1.jpg') }}" alt="Image 1">
<img src="{{ url_for('static', filename='images/sarc-2.jpg') }}" alt="Image 2">
<img src="{{ url_for('static', filename='images/sarc-5.png') }}" alt="Image 3">
<img src="{{ url_for('static', filename='images/sarc-4.jpg') }}" alt="Image 4">
<img src="{{ url_for('static', filename='images/sarc-6.jpg') }}" alt="Image 5">
</div>
</div>
<div class="content">
<h2>PROJECT TITLE : Detecting Sarcasm Across Headlines and Text</h2>
<p>
        This project is designed to detect sarcasm in text using a trained deep learning model which is CNN-BILSTM model.
        Our goal is to simplify the understanding of sarcastic headlines and text by leveraging AI.
</p>
<p>If you want to identify whether the given text is sarcastic or not, click the below button for prediction.</p>
<button onclick="location.href='/about'">Predict</button>


</div>
</div>
{% endif %}

{% if active_tab == 'about' %}
<div class="about-section">
<!-- <h2>About the Project</h2> -->
<p>This project uses deep learning models i.e, CNN-BiLSTM model to detect sarcasm in text.</p>
<div class="form-container">
```

```
<h3>Sarcasm Detection</h3>
<form action="/predict" method="POST" id="prediction-form">
<label for="text">Enter Text:</label>
<textarea id="text" name="text" required></textarea>
<button type="submit">Check</button>
</form>
<div id="result"></div>
</div>
</div>
{% endif %}

{% if active_tab == 'flowchart' %}
<div class="flowchart-section">
<h2>Project Flowchart</h2>
        <img src="{{ url_for('static', filename='images/kfold included.png') }}"
alt="Flowchart" width="1100" height="500">
</div>
{% endif %}
</div>
<!-- output refreshing -->
<script>
const form = document.getElementById('prediction-form'); const resultDiv =
document.getElementById('result'); const inputField =
document.getElementById('text');

// Clear result when user starts typing inputField.addEventListener('input', () => {
resultDiv.innerText = ''; // Clear previous result
});

// Handle formsubmission if (form) {
form.addEventListener('submit', async (event) => { event.preventDefault();

const formData = new FormData(form); const response = await fetch('/predict', {
method: 'POST', body: formData,
});
const result = await response.json(); if (result.error) {
resultDiv.innerText = result.error; resultDiv.style.color = 'red';
} else {
resultDiv.innerText = result.prediction;
resultDiv.style.color = result.prediction === 'Sarcastic' ? 'red' : 'blue';
}
});
}
</script>
</body>
```

```
</html>
```

Backend.py:

```
import tensorflow as tf import numpy as np import pickle

# Load the pre-trained model
model = tf.keras.models.load_model('./sarcasm_model.h5')

# Preprocess input for the model (update this based on your Colab preprocessing logic)
def preprocess_input(text):
# Example preprocessing - tokenize or vectorize the text (update as needed) # Replace
this with your own text preprocessing logic from Colab
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences


# These values must match those used during training
tokenizer = Tokenizer(num_words=10000, oov_token="<OOV>") # Load tokenizer
with open('tokenizer.pickle', 'rb') as handle: tokenizer = pickle.load(handle)
#tokenizer.fit_on_texts([""]) # Dummy fit (replace with training tokenizer config)

# Tokenize and pad input
sequences = tokenizer.texts_to_sequences([text])
padded = pad_sequences(sequences, maxlen=100, padding='post', truncating='post')
return padded

# Prediction function def predict(text):
if not text:
    return "Invalid input" try:
processed_text = preprocess_input(text) prediction = model.predict(processed_text)
    return "Sarcastic" if prediction >= 0.5 else "Not Sarcastic" except Exception as e:
return f"Error: {str€));
```

9 RESULT ANALYSIS

Here, we analyze the performance of our proposed models using different performance metrics. Five variants of different models were tested to find out which one performed the best for text sarcasm detection. To determine the most effective model for sarcasm detection, five different variants were tested. These models include a hybrid neural network combining CNN and LSTM with an attention mechanism, a Support Vector Machine (SVM), Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), and a Bidirectional LSTM combined with a Convolutional Neural Network (BILSTM+CNN). Each of these models was selected based on their ability to process sequential data and their effectiveness in natural language processing tasks. The evaluation of these models provided insights into their strengths and weaknesses in detecting sarcasm across different types of text.

These models include:

- Hybrid neural network: CNN and LSTM with the attention mechanism
- Support vector machine
- Long short-term memory
- Gated Recurrent Unit
- Bidirectional-LSTM + Convolutional Neural Network.

It is also evident that out of these five, BILSTM+CNN gave the best performance among all these models. It predicted the sarcastic news and general text rather well. On the other hand, other models were not quite working effectively well on both headlines and general text. The other models fail to predict sarcasm in general text with accuracy. This model BILSTM+CNN reached an overall accuracy of 97 percent. The integration of Bidirectional LSTM and CNN works together in capturing the essence of language contextualization. The model turns much more efficient for grasping sarcasm. The comparison of all the other models and their accuracies are mentioned in the Table:9.1 below.

Table: 9.1 Comparision of all models with their accuracy

MODELS	DATASETS	ACCURACY
Hybrid neural network	News Headlines	79%
SVM	News Headlines	77%
GRU	News Headlines	85%
LSTM	News Headlines	85%
CNN+BILSTM	News Headlines	93%
CNN+BILSTM	News Headlines + general text	97%

The provided graphs which is shown in Fig:9.1 illustrate the training and validation performance of the CNN-BiLSTM sarcasm detection model over multiple epochs. The left graph represents model accuracy, showing that training accuracy steadily increases, while validation accuracy fluctuates. This suggests that the model learns well on training data but may experience slight overfitting on validation data. The right graph represents model loss, where training loss decreases continuously, indicating effective learning. However, validation loss fluctuates instead of following a consistent downward trend, which further indicates possible overfitting. This suggests that while the model performs exceptionally well on seen data, its generalization to unseen data might need improvement, possibly by using regularization techniques such as dropout or early stopping.

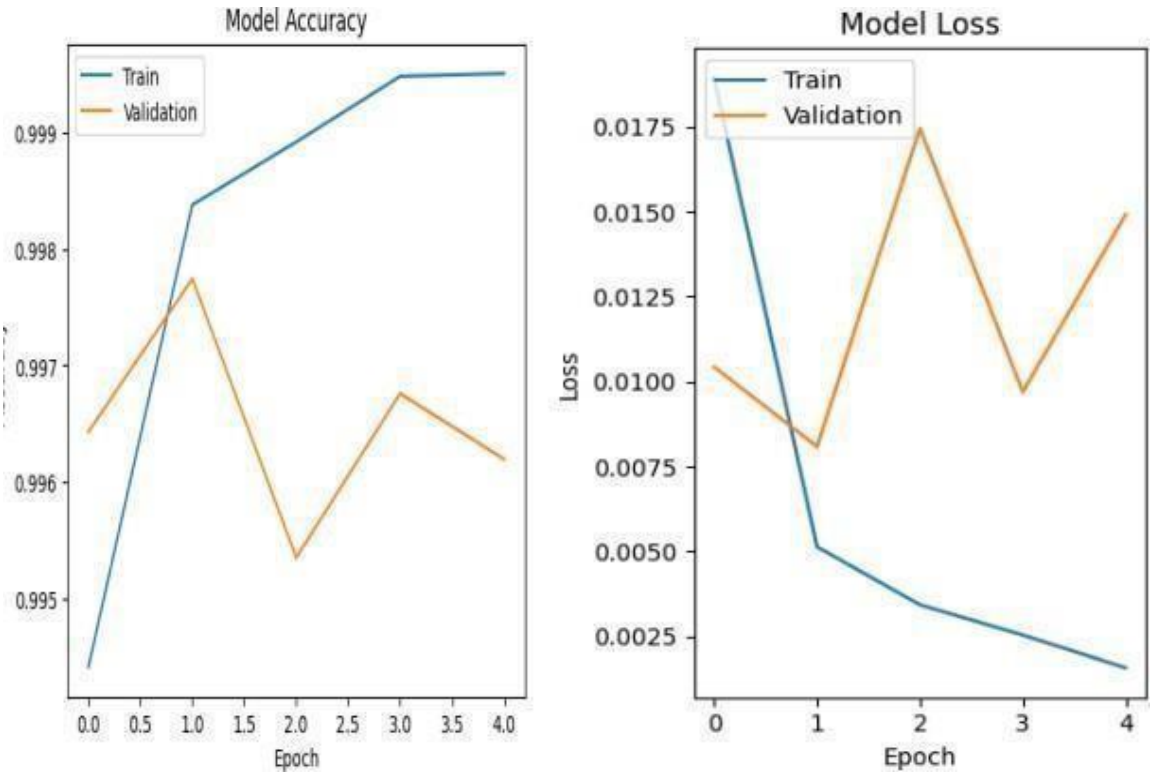

Fig : 9.1 Model Accuracy And Model Loss

```
Average Accuracy: 97.50569065411885%
Accuracy for fold 1: 93.87381672859192%
Accuracy for fold 2: 99.03290867805481%
Accuracy for fold 3: 99.61034655570984%
<ipython-input-28-fd050524e398>:14: UserWarning: \
  save_model(model, 'model/sarcasm_model.h5')
```

Fig: 9.2 Average accuracy of each 3 folds

The image which is Fig:9.2 displays the accuracy results obtained from a 3-fold cross- validation process for the sarcasm detection model. The accuracy values for the

three folds are 93.87%, 99.03%, and 99.61%, leading to an average accuracy of 97.50%. This indicates that the model generalizes well across different subsets of data. The variation in accuracy across folds suggests that some splits of the dataset may be slightly more challenging than others, but overall, the model maintains a consistently high performance. For the performance of these models, we measured the performances based on accuracy. In this experiment, we evaluated the model's performance by using accuracy and the F1- through epochs, it shows that the model gradually improves in predicting sarcastic text more correctly; simultaneously, accuracy increased linearly to touch an impressive 97% accuracy. Firstly, the similar trend in the validation loss and accuracy curves suggests that the model generalizes well on new, unseen data. As the model trains, the training-versus-validation gap in accuracy gets narrower, hence the model is not overfitting. Overall, the trends cut both ways to undergird the effectiveness of BILSTM+ text sarcasm detection. sarcastic text 97% of the time and non-sarcastic text 95% of the time. It does not make many mistakes: only 3% of sarcastic text is classified as non- sarcastic and 5% of non-sarcastic text.

The classification report provides an evaluation of the sarcasm detection model's performance based on precision, recall, and F1-score. The model achieves an overall accuracy of 98%, indicating a strong ability to distinguish between sarcastic and non-sarcastic text. For the "Not Sarcastic" class, the model attains 0.98 precision, 0.98 recall, and a 0.98 F1-score, signifying that most non-sarcastic headlines are correctly classified. Similarly, for the "Sarcastic" class, it achieves 0.97 precision, 0.97 recall, and a 0.97 F1-score, demonstrating its effectiveness in detecting sarcasm. The macro average (0.97) and weighted average (0.98) confirm that the model maintains a high performance across both classes. The support column indicates the number of samples in each category, showing a slightly higher number of non-sarcastic instances. Overall, these metrics suggest that the CNN-BiLSTM model is highly accurate and well-balanced in sarcasm classification.

The Fig:9.3 shows the Classification Report for Sarcasm Detection Model: The CNN-BiLSTM model achieves high accuracy (98%) with a precision of 0.98 for non-sarcastic and 0.97 for sarcastic labels. The balanced F1-score (0.97-0.98) indicates strong performance in detecting sarcasm effectively.

Classification Report:

	precision	recall	f1-score	support
Not Sarcastic	0.98	0.98	0.98	34470
Sarcastic	0.97	0.97	0.97	29434
accuracy			0.98	63904
macro avg	0.97	0.97	0.97	63904
weighted avg	0.98	0.98	0.98	63904

Fig: 9.3 Classification report of the model

The confusion matrix is a fundamental performance evaluation tool used to assess the outcomes of classification models. In the context of sarcasm detection, the confusion matrix provides detailed insights into the true negatives (TN), false positives (FP), and false negatives (FN) across sarcastic and non-sarcastic categories.

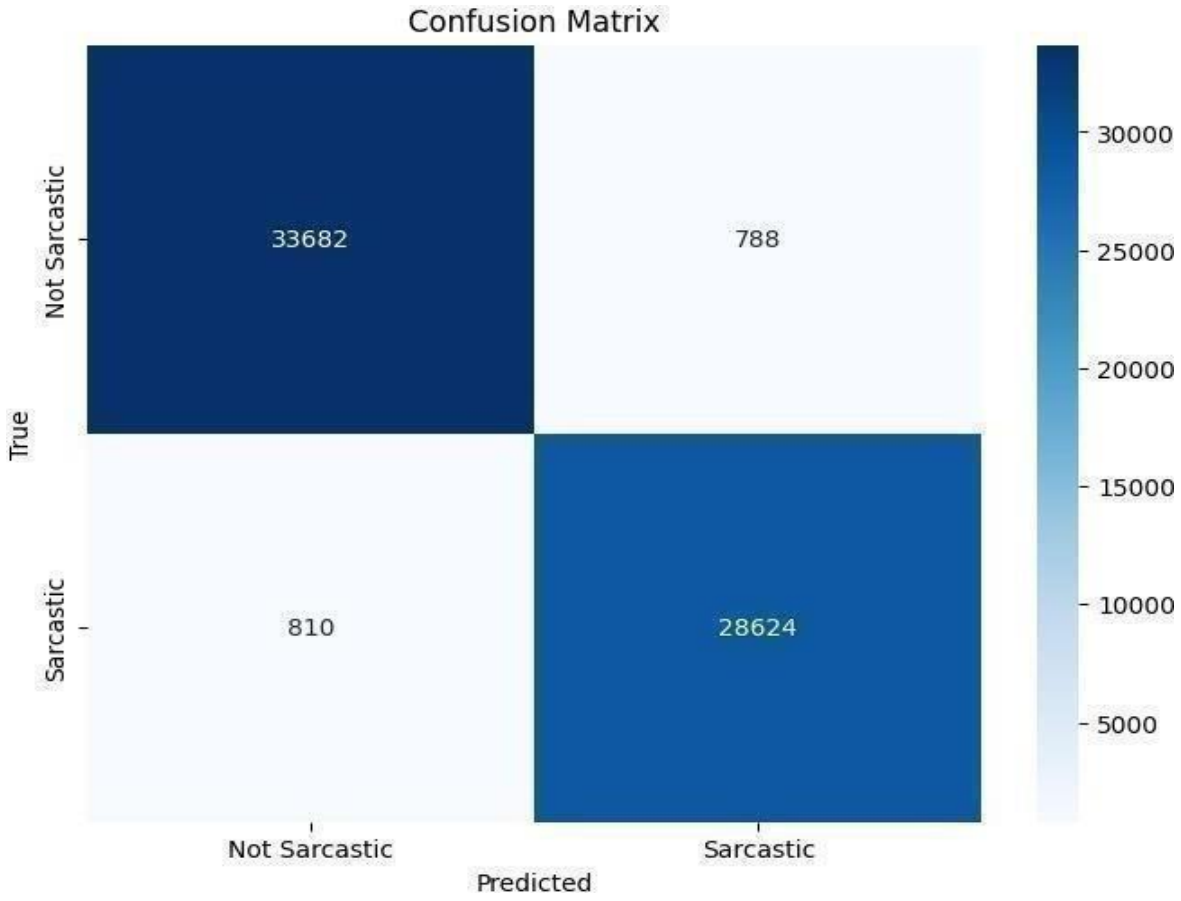

Fig: 9.4 Confusion Matrix

The above Fig:9.4 shows the Confusion Matrix for Sarcasm Detection Model.

The model correctly classifies 33,682 non-sarcastic and 28,624 sarcastic instances, with minimal misclassifications (788 false positives and 810 false negatives), demonstrating high accuracy in sarcasm detection.

This trends promisingly for the proposed model, BILSTM+CNN was reveals both from loss and accuracy graphs during training in Fig:4. While the loss decreased consistently through epochs, it shows that the model gradually improves in predicting sarcastic text more correctly; simultaneously, accuracy increased linearly to touch an impressive 97% accuracy. Firstly, the similar trend in the validation loss and accuracy curves suggests that the model generalizes well on new, unseen data. As the model trains, the training-versus-validation gap in accuracy gets narrower, hence the model is not overfitting. Overall, the trends cut both ways to undergird the effectiveness of BILSTM+CNN in text sarcasm detection.

10 TEST CASES

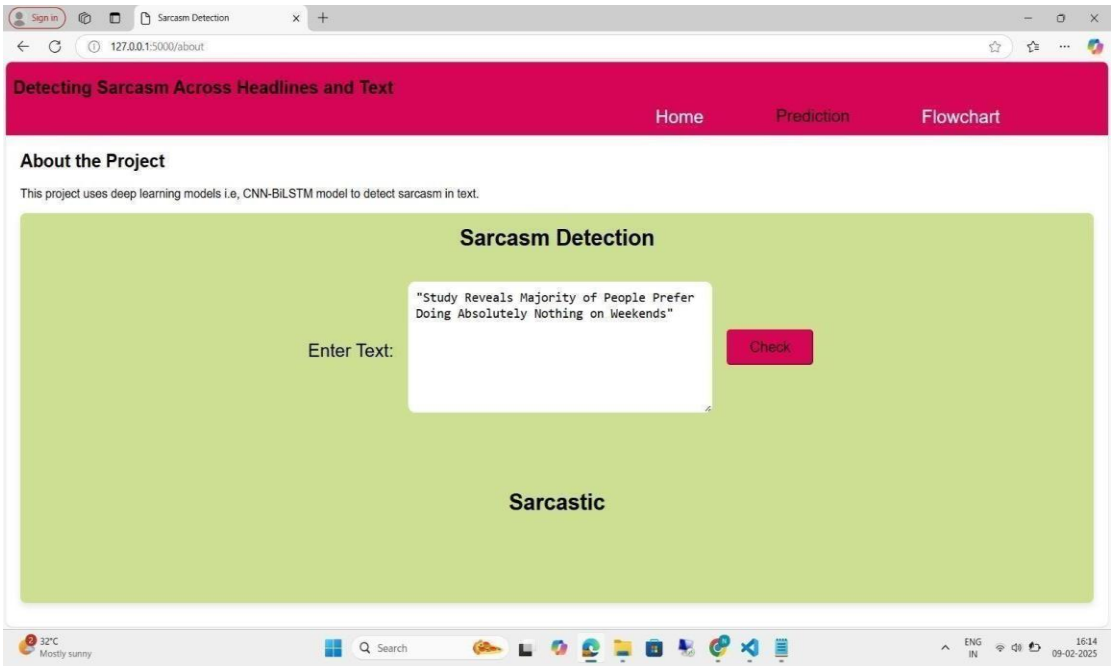

Fig :10.1 Headlines Prediction

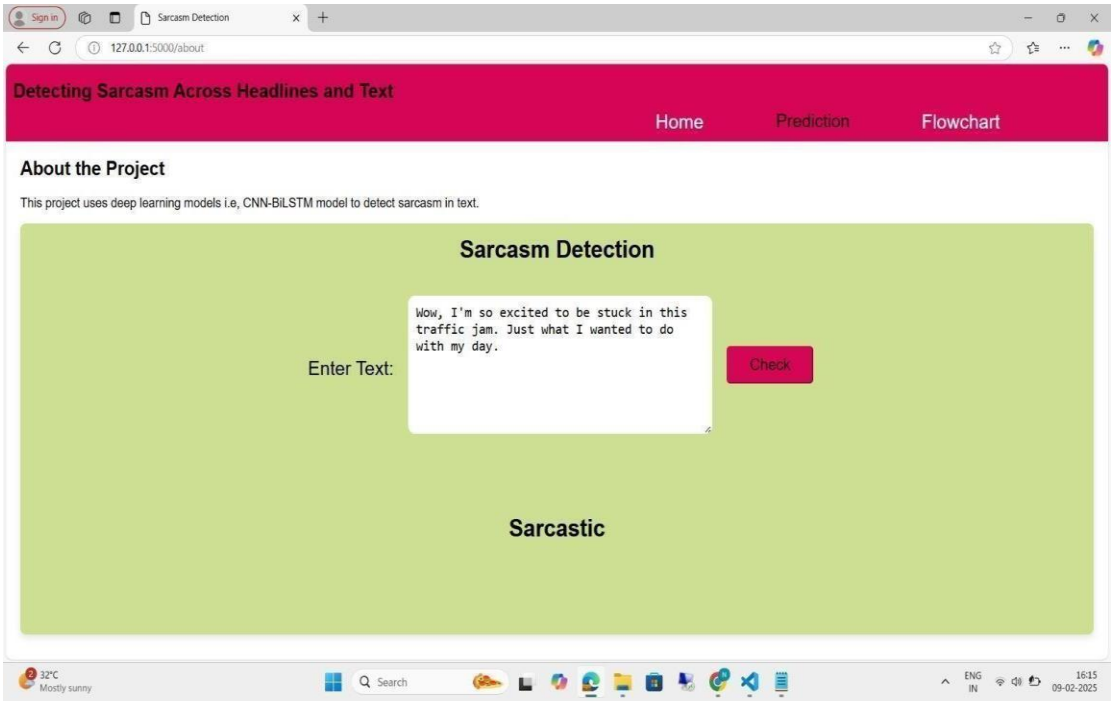

Fig: 10.2 General Text Prediction

The Figure 10.1 illustrates the prediction of headlines from the given dataset, showcasing the model's ability to detect sarcasm effectively. The CNN-BiLSTM model processes the textual data to differentiate between sarcastic and non-sarcastic headlines. This visualization helps analyze how well the model captures contextual and linguistic cues for sarcasm detection. The Figure 10.2 illustrates the prediction of general text when processed by the CNN-BiLSTM model for sarcasm detection. The model analyzes linguistic patterns and contextual features to determine whether the text is sarcastic or not. This visualization highlights the model's capability in handling diverse textual inputs effectively.

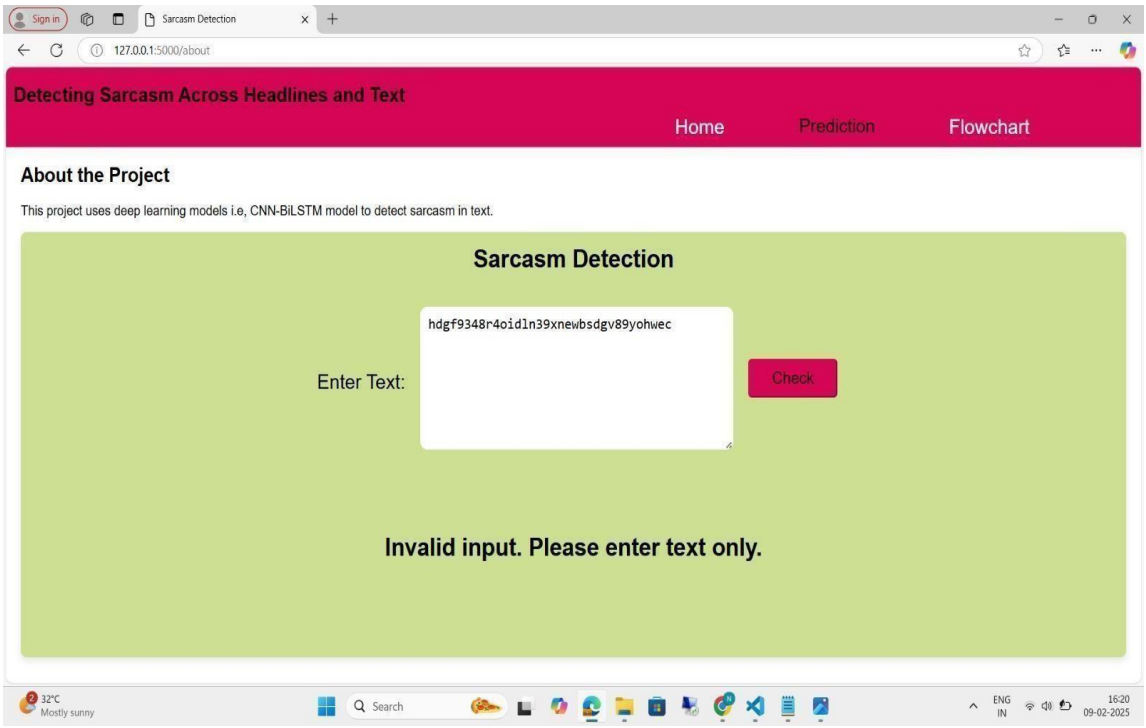

Fig :10.3 Checking validation of input text

The above Fig:10.3 describes about the validation of the text where it doesn't consider the random text, which demonstrates the validation process of input text in the sarcasm detection model. The CNN-BiLSTM model evaluates the given text to ensure accurate classification between sarcastic and non-sarcastic statements. This

step is crucial for assessing the model's reliability and performance in real-world scenarios.

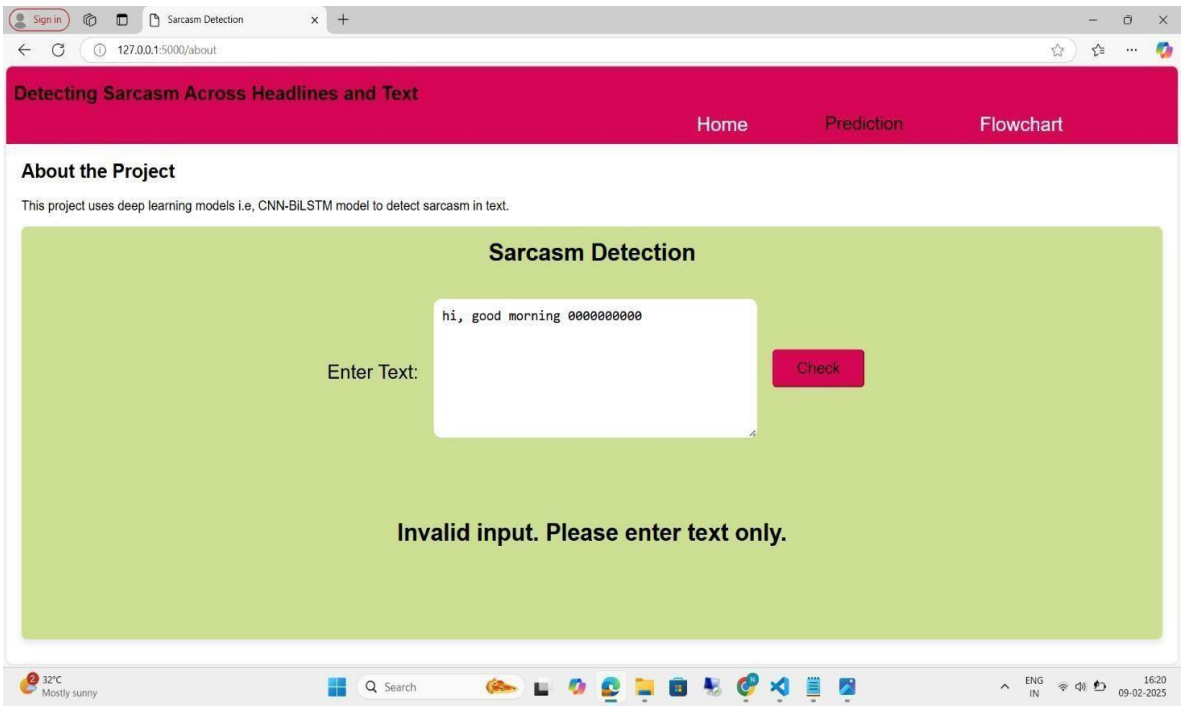

Fig: 10.4 Validation2 of input text

The above Fig:10.4 describes about the validation of the text where it doesn't consider the numbers in the text, which represents the second validation phase of the input text in the sarcasm detection model. This step further refines the classification process by verifying the accuracy of sarcasm detection. It ensures the model's consistency in distinguishing sarcastic and non-sarcastic text effectively.

11 OUTPUT SCREENS

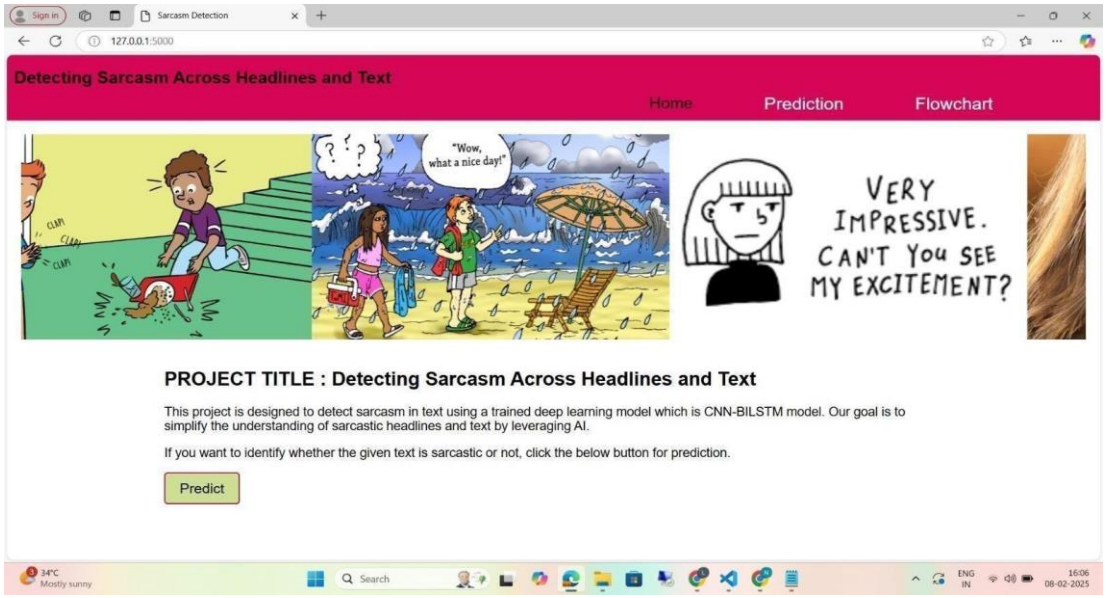

Fig: 11.1 Home Screen

The above Fig:11.1 describes about the main home page of our sarcasm detection website, which showcases the main homepage of our sarcasm detection website. It serves as the entry point where users can give text and receive real-time predictions. The design ensures accessibility for efficient sarcasm detection.

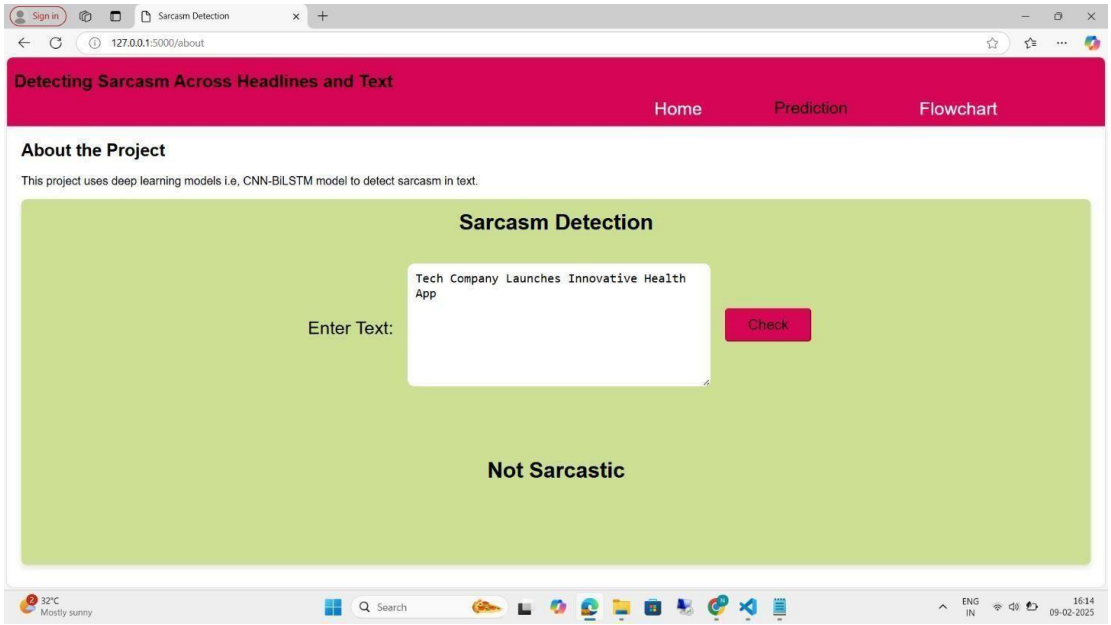

Fig: 11.2 prediction page

The above Fig:11.2 describes about the sarcasm detection tab in our website. It illustrates the sarcasm detection tab on our website, where users can enter text to check for sarcasm. This section processes the input using the CNN-BiLSTM model and displays whether the text is sarcastic or not. The tab provides a user-friendly interface for seamless sarcasm detection.

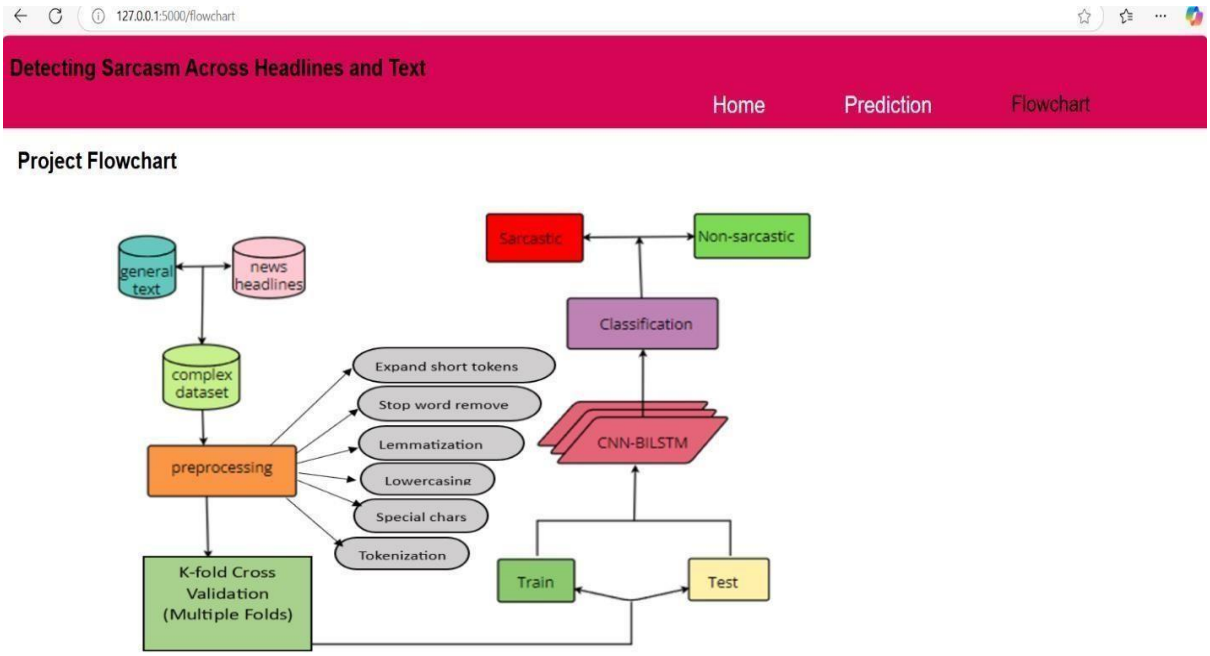

Fig:11.3 Flowchart of the Project

The above Fig:11.3 describes about the entire model working flowchart of our project, which presents the flowchart of the project, outlining the step-by-step process of sarcasm detection using the CNN-BiLSTM model. It visually represents data preprocessing, feature extraction, model training, and prediction phases.

12 CONCLUSION

To summarize, the proposed BILSTM+CNN has shown exceptional performance in text sarcasm detection. It attained an overall accuracy of 97%, outperforming other models such as LSTM, GRU, Hybrid neural network, and SVM. The major reasons for the efficacy of this model over others are its sensitivity to the context and subtleties of language that define what text is sarcastic. The confusion matrix shows high precision and recall values of both classes, indicating that very few errors have occurred. Loss and accuracy trends are indicative of the model's learning and generalization. The word cloud speaks to the variation in the usage of language in sarcastic and non-sarcastic text. The performance of the BILSTM+CNN model is consistent over a number of datasets; hence, the approach taken to detect sarcasm is reliable.

Here, the combined use of Headlines and Sarcasm datasets with 63904 texts, including headlines and sarcasm texts, may help the model get effectively trained to perform well in both seen and unseen data to detect sarcasm. This research encourages the development of more accurate natural language processing systems. These results have consequences in applications like sentiment analysis, text classification, and social media monitoring. Overall, the BILSTM+CNN model constitutes a significant leap in the detection of sarcasm and opens new avenues to future research in this area.

13 FUTURE SCOPE

Sarcasm detection in text, especially in news headlines, holds immense potential for future advancements. As social media and digital communication continue to grow, sarcasm detection can play a vital role in improving various natural language processing (NLP) applications such as sentiment analysis, opinion mining, and automated content moderation. Accurately identifying sarcasm in text can help AI systems understand human emotions better and improve their responses in different scenarios. A key area for improvement is multimodal sarcasm detection, which combines text analysis with voice tone, facial expressions, and gestures. This would allow AI to detect sarcasm more effectively in spoken conversations and videos. Additionally, adapting models for different languages and cultures is crucial since sarcasm varies across regions. Developing multilingual sarcasm detection systems will make AI applications more globally relevant.

Enhancing real-time sarcasm detection for chatbots and virtual assistants can improve human-computer interactions, making AI more responsive and context-aware. Another important application is misinformation detection, where sarcasm detection can help prevent sarcastic remarks from being misinterpreted as factual statements. Techniques such as self-supervised learning and attention mechanisms will help AI grasp sarcasm more effectively. In conclusion, sarcasm detection will continue to evolve, improving AI's ability to understand language, detect misinformation, and enhance user interactions. With advancements in multimodal analysis, real-time processing, and deep learning, AI can become more accurate, context-aware, and reliable in detecting sarcasm across different platforms.

14 REFERENCES

[1] Chy, M. S. R., Chy, M. S. R., Mahin, M. R. H., Rahman, M. M., Hossain, M. S., Rasel, A. (2023, November). Sarcasm Detection in News Headlines Using Evidential Deep Learning- Based LSTM and GRU. In Asian Conference on Pattern Recognition (pp. 194-202). Cham:Springer Nature Switzerland

[2] Misra, R.,Arora, P. (2023). Sarcasm detection using news headlines dataset. AI Open, 4, 13-18.

[3] Rafi, S., Das, R. Topic-guided abstractive multimodal summarization with multimodal output. Neural Comput and Applic (2023).https://doi.org/10.1007/s00521-023-08821-5

[4] Jayaraman, A. K., Trueman, T. E., Ananthakrishnan, G., Mitra, S., Liu, Q., Cambria, E. (2022, December). Sarcasm Detection in News Headlines using Supervised Learning. In 2022International Conference on Artificial Intelligence and Data Engineering (AIDE) (pp. 288- 294). IEEE.

[5] S. Rafi and R. Das, ”A Linear Sub-Structure with Co-Variance Shift for Image Captioning,” 2021 8th International Conference on Soft Computing and Machine Intelligence (ISCMI), Cario, Egypt, 2021, pp. 242-246, doi: 10.1109/ISCMI53840.2021.9654828.

[6] S. Rafi and R. Das, ”Abstractive Text Summarization Using Multimodal Information,” 2023 10th International Conference on Soft Computing and Machine Intelligence (ISCMI), Mexico City, Mexico, 2023, pp. 141- 145,doi: 10.1109/ISCMI59957.2023.10458505.

[7] Ali, R., Farhat, T., Abdullah, S., Akram, S., Alhajlah, M., Mahmood, A., Iqbal, M. A. (2023).
Deep learning for sarcasm identification in news headlines. Applied Sciences, 13(9), 5586

[8] S. Rafi and R. Das, ”RNN Encoder And Decoder With Teacher Forcing Attention Mechanism for Abstractive Summarization,” 2021 IEEE 18th India Council International Conference (INDICON), Guwahati, India, 2021, pp. 1-7, doi: 10.1109/INDICON52576.2021.9691681.

[9] Mohan, A., Nair, A. M., Jayakumar, B., Muraleedharan, S. (2023). Sarcasm detection us- ing bidirectional encoder representations from transformers and graph convolutional net- works. Procedia Computer Science, 218, 93-102

[10] Chudi-Iwueze, O., Afli, H. (2020). Detecting Sarcasm in News Headlines. In CERC (pp. 100-111).

[11] Suma, D., Raviraja Holla, M., Darshan Holla, M. (2024). Decoding sarcasm: unveiling nuances in newspaper headlines. International Journal of Electrical and Computer Engi-neering (IJECE), 14(3), 3011-3020

[12] Kaya, S., Alatas, B. (2022). Sarcasm detection with a new cnn+ bilstm hybrid neural net- work and bert classification model. International Journal of Advanced Networking and Applications, 14(3), 5436-5443

[13] B Naga, S., K Santhi, S., P Radha, M. (2023). A Deep Learning Approach for Sarcasm De- tection in User generated Content. Journal Of Technology, 11(12).

[14] 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

[15] Barhoom, A., Abu-Nasser, B. S., Abu-Naser, S. S. (2022). Sarcasm detection in headline news using machine and deep learning algorithms

[16] Azwar, A. S. (2020). Sarcasm detection using multi-channel attention based BLSTM on news headline.

[17] Helal, N. A., Hassan, A., Badr, N. L., Afify, Y. M. (2024). A contextual based approach for sarcasm detection. Scientific Reports, 14(1), 15415.

3rd IEEE International Conference on Interdisciplinary Approaches in Technology and Management for Social Innovation (IEEE IATMSI-2025)

CERTIFICATE OF AUTHORSHIP & ACCEPTANCE

This is to certify that Paper Titled

Detecting Sarcasm Across Headlines and Text

Submitted by Authors

Ambati Lakshmi Niharika, Seva Neelima, Kadiyala Nikhitha, Shaik Rafi(Narasaraopeta Engineering College - JNTUK)

is accepted and presented in IEEE International Conference on Interdisciplinary Approaches in Technology and Management for Social Innovation (IATMSI-2025), organized by IEEE MP Section and ABV-IIITM Gwalior, India on 6-8 March 2025.

Dr. Arun Kumar, Organizing Chair, IATMSI-2025

Prof. G. S. Tomar, General Chair, IATMSI-2025

Dr. Somesh Kumar, Conference Chair, IATMSI-2025

3rd IEEE International Conference on Interdisciplinary Approaches in Technology and Management for Social Innovation (IEEE IATMSI-2025)

CERTIFICATE OF PRESENTATION

This is to certify that

Ambati Lakshmi Niharika (Narasaraopeta Engineering College – JNTUK)

Presented a Paper Titled

Detecting Sarcasm Across Headlines and Text

in IEEE International Conference on Interdisciplinary Approaches in Technology and Management for Social Innovation (IATMSI-2025), organized by IEEE MP Section and ABV-IIITM Gwalior, India on 6-8 March 2025.

Dr. Arun Kumar, Organizing Chair, IATMSI-2025

Prof. G. S. Tomar, General Chair, IATMSI-2025

Dr. Somesh Kumar, Conference Chair, IATMSI-2025

Detecting Sarcasm Across Headlines and Text

Shaik Rafi
Assistant Professor, Dept of CSE
Narasaropeta Engineering Collage
Narasaraopet-522601, Andhra Pradesh, India.
shaikrafinrt@gmail.com

Ambati Lakshmi Niharika
Dept of CSE
Narasaropeta Engineering Collage
Narasaraopet, Palnadu, AP, India.
ambatinikki@gmail.com

Seva Neelima
Dept of CSE
Narasaropeta Engineering Collage
Narasaraopet, Palnadu, AP, India.
sevaneelima13@gmail.com

Kadiyala Nikhitha
Dept of CSE
Narasaropeta Engineering Collage
Narasaraopet,Palnadu,Ap,India
nikhitha.k234@gmail.com

Mothe Sathyam Reddy
Assistant Professor, Dept of CSE
Narasaropeta Engineering Collage
Narasaraopet,Palnadu,Ap,India
sathyamreddym@gmail.com

Moturi Sireesha
Associate Professor, Dept of CSE
Narasaropeta Engineering Collage
Narasaraopet,Palnadu,Ap,India
sireeshamoturi@gmail.com

***Abstract*—In this era with the rapid growth in social media usage among the current generation, a huge amount of content and comments, most of them sarcastic, is seen. Sarcasm has turned out to be an important part of daily life, especially in news and social media, where sarcastic comments are often used for better attention. However, detecting sarcasm is always challenging because it deals with understanding the difference between what has been said and what is meant. The current paper focuses on the detection of sarcasm in news headlines with the help of deep learning. Previous works were based on a wide range of datasets; however, these had limitations regarding either size or quality. In this respect, the authors propose creating a new dataset of headlines from sarcastic news sites and real news sites that is large and of high quality, hence appropriate for machine learning model training. The authors have also used the CNN-BILSTM architecture for text analysis, identifying sarcasm expression and deciding whether it is sarcastic or not-sarcastic which gained an accuracy of 97%. This dataset is made publicly available to enable further research in this direction.**

***Index Terms*—Sarcasm detection, News headlines dataset, Text, Deep learning, NLP, Convolutional Neural Network(CNN), BI-LSTM.**

I. Introduction

These days, the phone and technology are slowly and gradually becoming an inseparable part of our existence. We cannot even imagine a single day without social media browsing, video watching, or messaging. In turn, because of this, social media has become congested, and people express themselves more than ever. But with the rise of social media comes an increase in sarcastic comments and humorous postings. It would seem like each one of them had now become some kind of stand-up comedian online. Some use sarcasm to be funny or make a point, others to cover their feelings or even cruelly. Whichever way it goes, sarcastic comments are taking over in social media, and this presumably changes how we interact online. Sarcasm is a subtle way of expressing things where the words mean quite the opposite of their literal sense [2,3,4]. It is often employed to mock or to show contempt. It is widely used while speaking and in writing with the intention of drawing attention and stating one's opinion. The detection of sarcasm remains a challenging task, as it is by nature ambiguous in nature and depends on context and common sense knowledge [14]. Hence, even humans and machines find it difficult to correctly detect the presence of sarcasm.

The need for sarcasm detection in NLP is immense in the present digital age, since a huge amount of content is being produced on online platforms such as social media and news websites. Most of the informal languages with contextual references end up in noisy datasets, which in turn makes the process of detection tough. Also, the labeling of sarcasm in datasets manually is a time-consuming process and may lead to inconsistencies in many cases due to different levels of interpretation of sarcasm as seen in [2,16]. Research in sarcasm detection has applied several different techniques: rule-based approaches, machine learning, and deep learning. The collection of social media datasets is typically performed through tag-based supervision, which is both error-prone and limited with regard to vocabulary as mentioned in [7,8]. At the same time, high-quality, manually labeled datasets are rather small in size and too costly to produce, which results in underpowered models. More overwhelming is the fact that models have to comprehend the subtlety and contextual dependency of sarcastic language. While NLP has come a long way, most of the models lack this subtlety in sarcasm and instead depend on lexical cues rather than actual comprehension. While sarcasm detection keeps improving, new challenges have opened up for researchers in multi-modal analysis that is, considering both the visual and audio cues that would give a better understanding of the nuances of sarcastic expression. The

979-8-3315-2169-1/25/$31.00 ©2025 IEEE

so much on context. Different methods are debated in the paper as in [13] and [14] from different angles but with a strong support for feature extraction techniques as an indispensable block in improving model performance. It provides a framework to improve the systems on the detection of sarcasm by analyzing the different approaches that exist and giving a proposal for future directions to reach the solution. Decoding Sarcasm: Unveiling Nuances in Newspaper Headlines"[9] - a paper on challenges for sarcasm detection in NLP based on newspaper headlines - very effectively situates this challenge. It is the subtle contextual cues flipping the literal meaning of words that make headline sarcasm hard to detect. This paper elaborates on the different feature extraction and modeling techniques the authors tried, setting up context to a few nuances of sarcastic speech as same in [12]. Work acts like an eye-opener in improving the models for sarcasm detection. It opens the doors to much more accurate and advanced systems. O.Chudi-Iwueze, H.Afli, of "Sarcasm Detection with a New CNN+BiLSTM Hybrid Neural Network and BERT Classification Model" addresses the challenge of detecting sarcasms in social media-a place where sarcastic expressions occur quite often and have mostly been misinterpreted in NLP[10]. They also propose a hybrid model that includes CNN, BiLSTM, and BERT to enhance the efficiency of sarcasm detection as similar to [11]and[15]. It handles the intricateness of sarcastic language through a model that provides informal, contextual communication online to reduce misunderstandings and improve the capabilities of NLP systems in social media interaction analysis.

III. Proposed Work

A. Preprocessing

Among all, the preprocessing of textual data is one of the most important steps in getting the data prepared for effective analysis and model training in natural language processing that involves processing of raw text data into a form appropriate for model training and evaluation. Some of the techniques that find their employment here include special character handling, lowercasing, stopword removal, expanded shortened tokens, tokenization, and lemmatization. All these preprocessing techniques work in concert to transform raw text into a cleaner and more consistent format, which greatly enhances the performance of NLP models by better grasping the underlying semantic information contained within.

1) Data Cleaning: Objective: This removes the noise and irrelevant information so that the model focuses on what is meaningful from the text. • Handling special characters: Special character handling in deep learning refers to the process of removing or replacing non-alphanumeric characters in text to clean and standardize input data for improvement in performance of models. • Lowercasing: In deep learning, lowercasing represents a pre-processing strategy in text where all characters are converted to lowercase in order to reduce variability and thereby improve model performance. • Stopword removal: In deep learning, stopword removal refers to the removal of common words such as "the," "and," and "is" from text in order to reduce the noise and have the model focus on other, more meaningful words while training. • expanding shortend tokens: The shortening-end token expansion is a preprocessing NLP technique in which the tokens of small size, which could be words or sub-words, get transformed into more informative and helpful tokens for later tasks such as text classification or language modeling. • Tokenization: Have a look at how you tokenized the text; for example, into words or subwords. • Lemmatization: Lemmatization in deep learning refers to the reduction of inflectional forms by converting words into their base form or root and improvise consistency for model training. The Table:1 is representative of text before and after performing all the preprocessing techniques mentioned above , where the "Text-before preprocessed" column presents the text in raw form-that is, noisy, punctuation, special character-containing form-whereas the "Text-after preprocessed" column contains the text after different techniques of preprocessing have been applied to remove or correct such issues.

TABLE I
HEADLINES BEFORE AND AFTER PREPROCESSING

Text-before preprocessed	Text-after preprocessed
1.mother ferries 4 more shirt options back to son in gap dressing room	1. mother ferry 4 shirt option back son gap dressing room
2. zoo animals roam free after flooding in Tbilisi	2. zoo animal roam free flooding tbilisi
3. leave no person with disabilities behind	3. leave person disability behind
4. my disastrous search for the perfect swim-suit	4. disastrous search perfect swim-suit
5. my white inheritance	5. white inheritance

The Fig:1 represents word-clouds of given text or headlines, indicating a clear difference in the language used in each type of headline to visually represent the tone and sentiment differences in text. Words are set out in a cloud-like pattern, with the most frequently used words appearing in larger font sizes. These word-clouds highlight the variation in the usage of words appearing in sarcastic versus non-sarcastic headlines and visually understand the tone and sentiment in text. The words are set out in a cloud-like pattern; the more frequently used words are in bigger font sizes.

Fig. 1. Word clouds for sarcastic versus non-sarcastic headlines

B. Algorithm

Model Architecture: Sarcasm detection in the news headline is a complicated task; hence, CNN-BiLSTM forms one of

the promising deep learning approaches. It captures both local and global features using two techniques, converts headlines into numerical code, extracts features, analyzes context, and relationships. This model will get the probability of sarcasm and is trained on labeled headlines. Though effective, the biggest drawback it faces is related to the need for data and computational expense. This can further be improved by ensembling, transfer learning, and data augmentation. This technology could assist readers in tone and intent to make a much more aware public and hopefully decrease misinformation on social media and online forums. The CNN-BiLSTM model shows very good results in the detection of sarcasm from news headlines. First, it represents headlines as word embeddings that capture semantic relationships and then extracts local features such as n-grams and sentiment-bearing expressions using CNNs. The BiLSTM network will pick up the contextual dependencies and subtle patterns within this word sequence. Further, this will allow the model to extract even complex sarcasm cues, such as ironies and understatement which is also shown in the reference[3]. It is fed into a fully connected layer to yield a probability score on sarcasm detection. Therefore, the model will arrive at robust and highly accurate sarcasm detection, considering that CNNs are integrated along with the BiLSTMs, where the workflow is represented in a flowchart i.e, Fig.3.

IV. Experiment Setup

The main goal of the experiment is to identify sarcasm in the given headlines with the effective detection of sarcasm on general text. Here, the models are trained using publicly available datasets from the Kaggle website by the author RahulMisra. The models are trained and tested on Google Colab, a cloud-based environment that provides free access to GPUs, hence suitable for deep learning tasks. These cleaned and preprocessed text data were free of noisy data and prepared the ground for model training.

A. Dataset

In our study, we employed two datasets for sarcasm detection. First, the headlines dataset was very carefully selected in order to avoid issues such as wrong labeling and other language problems. Besides that, the fact that such headlines are written by professional journalists adds another level of reliability in the dataset. Some common problems of many user-generated datasets-such as slang, misspellings, and informal language-introduce noise, generally making the task of a model harder. But in this case, the formal structure of the language served for the model to pay attention just to the nuances of sarcasm itself and not to be distracted by irrelevant text variations. It contains sarcastic news headlines from the satirical website "The Onion"(https://www.theonion.com/), which tends to publish humoristic or ironical materials. As a counterpoint, nonsarcastic real news headlines were collected from "The HuffPost" (https://www.huffpost.com/), a popular news website famous for its high professional standards in regard to publications. Since both sets of headlines are in formal language, written by professional writers, the possibility of spelling errors or using informal language is very low. By using only sarcastic content from "The Onion," we ensure that the quality of our data is going to be high since the labels for being sarcastic will, in fact, be accurate. This reduces the chances of wrongly labeled headlines, which is a very important feature in detecting sarcasm since correct labeling will make the model work effectively. For the sarcasm detection task, we made use of a dataset from Kaggle named News Headlines Dataset for Sarcasm Detection https://www.kaggle.com/datasets/rmisra/news-headlines-dataset-for-sarcasm-detection. There used to be two versions: Version 1 and Version 2. Version 1 comprises headlines numbering about 26,709, with 11,724 labeled as sarcastic and 14,985 labeled as non-sarcastic. Version 2 comprises 28,619 head-lines of which 13,634 are sarcastic whereas the remaining 14,985 are not sarcastic. Every version consists of three columns, "headline" for the text of the news headline, "article-link" as an optional column, can be used for reference, and "is-sarcastic" is a binary label. 1 means the headline is sarcastic, while 0 otherwise, which are shown in Table:II. To this end, we created a count-plot showing the number of both sarcastic and non-sarcastic headlines. In the plot below, Fig:2, 0 represents non-sarcastic headlines and 1 represents sarcastic head-lines. This chart gives us insight on the balance between the two types of head-lines in both versions of the dataset.

TABLE II
COLUMNS IN THE NEWS HEADLINES DATASET AND THEIR DESCRIPTION

Column	Description
headline	It has the headline text which we use later for sarcasm detection.
article_link	It has the link for the article from which the headline was taken. It was given as a reference
is_sarcastic	It has binary values as 0,1 where 1-represents the text as sarcastic and 0-represents the text as non-sarcastic

Also taken into account are the sarcasm dataset and both the version1 and version2 datasets, which are general texts consisting of 8576 general texts. These contain both sarcastic and non-sarcastic texts. The sarcasm dataset, i.e., general text, contains columns as "text"-which has sarcastic and non-sarcastic texts-and "Y"-output label has 0, 1 as values, which are shown in Table:III. In this case also, sarcastic news headlines are taken from the "The Onion" website, and non-sarcastic news headlines will be collected from "The HuffPost" website. The main reason for considering the sarcasm dataset with version1 and version2 is that while training the model with only headlines datasets, the model would be able to find out the sarcasm only on the headlines that are seen data, not the general text, which will be the unseen data. This is because, for better performance and for making correct predictions, the model has to be trained with both headlines and general sarcastic text. To do so, the sarcasm dataset is taken into consideration, which would be used for training

the model to make it perform effectively on the headlines and general text. Therefore, in total, 63904 lines of texts from headlines and general texts are combined, i.e., the entire dataset.

TABLE III
COLUMNS IN THE SARCASM DATASET AND THEIR DESCRIPTION

Column	Description
text	It has the general sarcastic and non-sarcastic texts.
Y	It has values of 0 (non-sarcastic) and 1 (sarcastic).

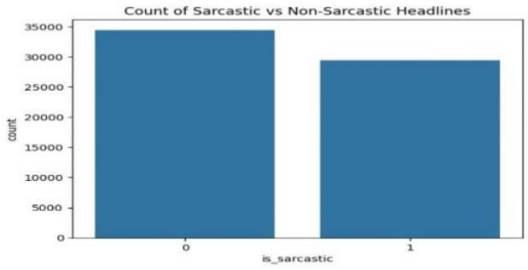

Fig. 2. Count-plot of Sarcastic vs Non-sarcastic headlines.

The Fig:2 indicates that among 63904 texts there are total of 34000 non-sarcastic texts and 29,904 sarcastic texts are present.

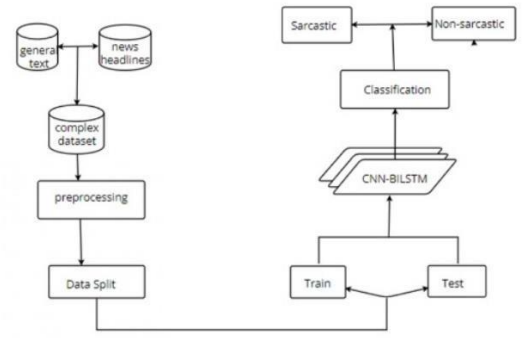

Fig. 3. Flowchart to represent the entire working of the model.

B. Training Environment

All the models have been executed on Google Colab since it provides a suitable environment to run such a computationally expensive task. In Colab, support for GPU has heavily accelerated the training processes of the models; thus, this allowed us to try out different models and hyperparameters in an efficient way.

C. Evaluation measures

For the performance of these models, we measured the performances based on accuracy. In this experiment, we evaluated the model's performance by using accuracy and the F1-score from the classification report, together with the confusion matrix to get the effectiveness of the model.

V. Result

Here, we analyze the performance of our proposed models using different performance metrics. Five variants of different models were tested to find out which one performed the best for text sarcasm detection. These models include: 1. Hybrid neural network: CNN and LSTM with the attention mechanism 2. Support vector machine 3. Long short-term memory 4. Gated Recurrent Unit 5. Bidirectional-LSTM + Convolutional Neural Network It is also evident that out of these five, BILSTM+CNN gave the best performance among all these models. It predicted the sarcastic news and general text rather well. On the other hand, other models were not quite working effectively well on both headlines and general text. The other models fail to predict sarcasm in general text with accuracy. This model BILSTM+CNN reached an overall accuracy of 97 percent. The integration of Bidirectional LSTM and CNN works together in capturing the essence of language contextualization. The model turns much more efficient for grasping sarcasm.

The comparison of all the other models and their accuracies are mentioned in the table:IV below.

TABLE IV
RESULTS AND COMPARISON OF ALL MODELS WITH THEIR ACCURACY

MODELS	DATASETS	ACCURACY
Hybrid neural network [2]	News Headline	79%
SVM	News Headline	77%
GRU	News Headline	85%
LSTM	News Headline	85%
CNN+BILSTM	News Headline	93%
CNN+BILSTM	News Headline + general text	97%

This trends promisingly for the proposed model, BILSTM+CNN was reveals both from loss and accuracy graphs during training in Fig:4. While the loss decreased consistently through epochs, it shows that the model gradually improves in predicting sarcastic text more correctly; simultaneously, accuracy increased linearly to touch an impressive 97% accuracy. Firstly, the similar trend in the validation loss and accuracy curves suggests that the model generalizes well on new, unseen data. As the model trains, the training-versus-validation gap in accuracy gets narrower, hence the model is not overfitting. Overall, the trends cut both ways to undergird the effectiveness of BILSTM+CNN in text sarcasm detection. The confusion matrix is telling us that the model correctly identifies sarcastic text 97% of the time and non-sarcastic text 95% of the time. It does not make many mistakes: only 3% of sarcastic text is classified as non-sarcastic and 5% of non-sarcastic text are misclassified as sarcastic as shown in Fig:5. The overall

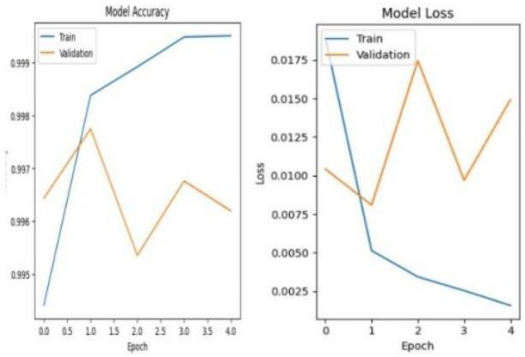

Fig. 4. Trend of loss and accuracy for the proposed method

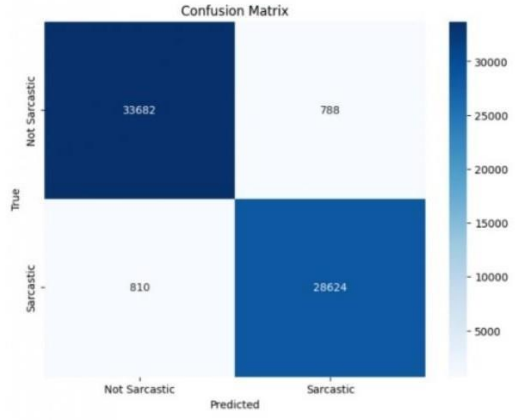

Fig. 5. confusion matrix of BI-LSTM + CNN model

confusion matrix is indicative of the high precision and recall, both from BILSTM+CNN, as proof of detection of sarcasm within text quite effectively.

VI. Conclusion

To summarize, the proposed BILSTM+CNN has shown exceptional performance in text sarcasm detection. It attained an overall accuracy of 97%, outperforming other models such as LSTM, GRU, Hybrid neural network, and SVM. The major reasons for the efficacy of this model over others are its sensitivity to the context and subtleties of language that define what text is sarcastic. The confusion matrix shows high precision and recall values of both classes, indicating that very few errors have occurred. Loss and accuracy trends are indicative of the model's learning and generalization. The word cloud speaks to the variation in the usage of language in sarcastic and non-sarcastic text. The performance of the BILSTM+CNN model is consistent over a number of datasets; hence, the approach taken to detect sarcasm is reliable. Here, the combined use of Headlines and Sarcasm datasets with 63904 texts, including headlines and sarcasm texts, may help the model get effectively trained to perform well in both seen and unseen data to detect sarcasm. This research encourages the development of more accurate natural language processing systems. These results have consequences in applications like sentiment analysis, text classification, and social media monitoring. Overall, the BILSTM+CNN model constitutes a significant leap in the detection of sarcasm and opens new avenues to future research in this area.

References

[1] Chy, M. S. R., Chy, M. S. R., Mahin, M. R. H., Rahman, M. M., Hossain, M. S., Rasel, A. A. (2023, November). Sarcasm Detection in News Headlines Using Evidential Deep Learning-Based LSTM and GRU. In Asian Conference on Pattern Recognition (pp. 194-202). Cham: Springer Nature Switzerland.

[2] Misra, R.,Arora, P. (2023). Sarcasm detection using news headlines dataset. AI Open, 4, 13-18.

[3] Rafi, S., Das, R. Topic-guided abstractive multimodal summarization with multimodal output. Neural Comput and Applic (2023). https://doi.org/10.1007/s00521-023-08821-5

[4] Jayaraman, A. K., Trueman, T. E., Ananthakrishnan, G., Mitra, S., Liu, Q., Cambria, E. (2022, December). Sarcasm Detection in News Headlines using Supervised Learning. In 2022 International Conference on Artificial Intelligence and Data Engineering (AIDE) (pp. 288-294). IEEE.

[5] S. Rafi and R. Das, "A Linear Sub-Structure with Co-Variance Shift for Image Captioning," 2021 8th International Conference on Soft Computing and Machine Intelligence (ISCMI), Cario, Egypt, 2021, pp. 242-246, doi: 10.1109/ISCMI53840.2021.9654828

[6] S. Rafi and R. Das, "Abstractive Text Summarization Using Multimodal Information," 2023 10th International Conference on Soft Computing and Machine Intelligence (ISCMI), Mexico City, Mexico, 2023, pp. 141-145,doi: 10.1109/ISCMI59957.2023.10458505.

[7] Ali, R., Farhat, T., Abdullah, S., Akram, S., Alhajlah, M., Mahmood, A., Iqbal, M. A. (2023). Deep learning for sarcasm identification in news headlines. Applied Sciences, 13(9), 5586

[8] S. Rafi and R. Das, "RNN Encoder And Decoder With Teacher Forcing Attention Mechanism for Abstractive Summarization," 2021 IEEE 18th India Council International Conference (INDICON), Guwahati, India, 2021, pp. 1-7, doi: 10.1109/INDICON52576.2021.9691681.

[9] Mohan, A., Nair, A. M., Jayakumar, B., Muraleedharan, S. (2023). Sarcasm detection using bidirectional encoder representations from transformers and graph convolutional networks. Procedia Computer Science, 218, 93-102

[10] Chudi-Iwueze, O., Afli, H. (2020). Detecting Sarcasm in News Headlines. In CERC (pp. 100-111).

[11] Suma, D., Raviraja Holla, M., Darshan Holla, M. (2024). Decoding sarcasm: unveiling nuances in newspaper headlines. International Journal of Electrical and Computer Engineering (IJECE), 14(3), 3011-3020

[12] Kaya, S., Alatas, B. (2022). Sarcasm detection with a new cnn+bilstm hybrid neural network and bert classification model. International Journal of Advanced Networking and Applications, 14(3), 5436-5443.

[13] B Naga, S., K Santhi, S., P Radha, M. (2023). A Deep Learning Approach for Sarcasm Detection in User generated Content. Journal Of Technology, 11(12).

[14] 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

[15] Barhoom, A., Abu-Nasser, B. S., Abu-Naser, S. S. (2022). Sarcasm detection in headline news using machine and deep learning algorithms

[16] Azwar, A. S. (2020). Sarcasm detection using multi-channel attention based BLSTM on newsheadline.

[17] Helal, N. A., Hassan, A., Badr, N. L., Afify, Y. M. (2024). A contextual-based approach for sarcasm detection. Scientific Reports, 14(1), 15415.

Detecting_Sarcasm_Across_Headlines_and_Text.pdf

ORIGINALITY REPORT

7%	4%	5%	2%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Nivin A. Helal, Ahmed Hassan, Nagwa L. Badr, Yasmine M. Afify. "A contextual-based approach for sarcasm detection", Scientific Reports, 2024 Publication	1%
2	Submitted to Riga Technical University Student Paper	1%
3	backoffice.biblio.ugent.be Internet Source	<1%
4	Md. Shamsul Rayhan Chy, Md. Shamsul Rahat Chy, Mohammad Rakibul Hasan Mahin, Mohammad Muhibur Rahman et al. "Chapter 15 Sarcasm Detection inNews Headlines Using Evidential Deep Learning-Based LSTM andGRU", Springer Science and Business Media LLC, 2023 Publication	<1%
5	content.sciendo.com Internet Source	<1%
6	www.journal.esrgroups.org Internet Source	<1%
7	i-scholar.in Internet Source	<1%
8	Submitted to Jose Rizal University Student Paper	<1%
9	Submitted to Liverpool John Moores University Student Paper	<1%

10	Nafisah Khalid, Nur Aina Shahrol. "Evaluation The Accuracy of Oil Palm Tree Detection Using Deep Learning and Support Vector Machine Classifiers", IOP Conference Series: Earth and Environmental Science, 2022 Publication	<1 %
11	"The Future of Artificial Intelligence and Robotics", Springer Science and Business Media LLC, 2024 Publication	<1 %
12	www.ijert.org Internet Source	<1 %
13	"Pattern Recognition", Springer Science and Business Media LLC, 2025 Publication	<1 %
14	arxiv.org Internet Source	<1 %
15	Abhilasha Sharma, Abhi Uday Pandey, Apoorv Gupta. "Sarcasm Detection on News Headline Dataset Using Language Models", 2023 3rd International Conference on Intelligent Technologies (CONIT), 2023 Publication	<1 %
16	Aditya Joshi, Pushpak Bhattacharyya, Mark J. Carman. "Investigations in Computational Sarcasm", Springer Science and Business Media LLC, 2018 Publication	<1 %
17	studentsrepo.um.edu.my Internet Source	<1 %
18	www.mdpi.com Internet Source	<1 %
19	"Computational Linguistics", Springer Science and Business Media LLC, 2018 Publication	<1 %

20	Dushyant Singh Chauhan, Gopendra Vikram Singh, Aseem Arora, Asif Ekbal, Pushpak Bhattacharyya. "An emoji-aware multitask framework for multimodal sarcasm detection", Knowledge-Based Systems, 2022 Publication	<1 %
21	QunHui Zhou, Tijian Cai. "Adaptive gate residual connection and multi-scale RCNN for fake news detection", Machine Learning with Applications, 2025 Publication	<1 %
22	Rasikh Ali, Tayyaba Farhat, Sanya Abdullah, Sheeraz Akram, Mousa Alhajlah, Awais Mahmood, Muhammad Amjad Iqbal. "Deep Learning for Sarcasm Identification in News Headlines", Applied Sciences, 2023 Publication	<1 %
23	discovery.researcher.life Internet Source	<1 %
24	link.springer.com Internet Source	<1 %
25	www.nature.com Internet Source	<1 %
26	Rajnish Pandey, Abhinav Kumar, Jyoti Prakash Singh, Sudhakar Tripathi. "A hybrid convolutional neural network for sarcasm detection from multilingual social media posts", Multimedia Tools and Applications, 2024 Publication	<1 %

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	On		