National College of Ireland

Deciphering Sarcasm in Textual Data: A Comparative Study of Machine Learning and Deep Learning Methods and a Nuanced Dive into Topic Modeling

MSc Research Project
Data Analytics

Ashlyn Nivita Mahendran Joseph Solomon
Student ID: x22163549

School of Computing
National College of Ireland

Supervisor: Shubham Subhnil

National College of Ireland
Project Submission Sheet
School of Computing

National College of Ireland

Student Name:	Ashlyn Nivita Mahendran Joseph Solomon
Student ID:	x22163549
Programme:	Data Analytics
Year:	2023
Module:	MSc Research Project
Supervisor:	Shubham Subhnil
Submission Due Date:	14/12/2023
Project Title:	Deciphering Sarcasm in Textual Data: A Comparative Study of Machine Learning and Deep Learning Methods and a Nuanced Dive into Topic Modeling
Word Count:	7,570
Page Count:	22

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	Ashlyn Nivita Mahendran Joseph Solomon
Date:	12th December 2023

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Deciphering Sarcasm in Textual Data: A Comparative Study of Machine Learning and Deep Learning Methods and a Nuanced Dive into Topic Modeling

Ashlyn Nivita Mahendran Joseph Solomon
x22163549

Abstract

In the landscape of digital communication, accurately identifying sarcasm presents a unique and complex challenge. The main objective of this study is to address this challenge by exploring the integration of advanced deep learning architectures in Natural Language Processing (NLP), aiming to enhance the precision and context-awareness in sarcasm detection within specialized textual domains. The dataset selected for this research consists of news headlines from two professional sources, The Onion and HuffPost, offering a distinct advantage of being free of noise due to their structured and professional journalistic standards. Latent Dirichlet Allocation (LDA) was initially employed for Topic Modeling to categorize the news headlines. Due to the low coherence score of 0.4418, the features extracted by LDA were found to be irrelevant for sarcasm detection, offering no practical utility in this specific analytical context. For sarcasm detection a robust approach was adopted by developing a hybrid deep learning model. This model combines Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN) with Bidirectional Long Short-Term Memory (BiLSTM) and Gated Recurrent Units (GRU), alongside GloVe embeddings. Tested over 10 epochs, this model achieved an accuracy of 82.19% on the test data, outperforming traditional machine learning models like Random Forest and Decision Tree, which recorded accuracies of 78.97% and 67.61% respectively.

1 Introduction

Digital communication is a rapidly evolving domain and accurately understanding the subtle aspects of language, such as sarcasm, has become extremely important. Since sarcasm is a combination of irony and humor (Gupta et al. (2020)), it raises a unique challenge in text-based communication. The accurate detection of sarcasm is essential for tasks such as content moderation, sentiment analysis and improving human-computer interaction. As digital platforms become more central to how communication occurs, understanding subtle language details is not only a technical issue but also crucial for making the online interactions clearer and empathetic. This is where Natural Language Processing (NLP) comes into picture. It has emerged as a pivotal technology for interpreting the complexities of human language (Mukherjee and Bala (2017)). NLP enables computers to understand, interpret, and respond to human language in a meaningful way, making it a valuable tool in the digital world. It plays a crucial role in a variety of

applications, from automated customer service chat-bots to sentiment analysis in social media.

The main objective of sarcasm detection is to determine whether a sentence is sarcastic or not. In recent years, a considerable amount of research has been done within the realm of Natural Language Processing (NLP) mainly focusing on sarcasm detection using both deep learning and machine learning techniques (Potamias et al. (2020)). These efforts have advanced the understanding of sarcasm in textual formats. To correctly understand sarcasm, it is often necessary to have a thorough knowledge of various kinds of information, such as what is being said, the situation in which it is said, and often, certain facts about the real world(Ghaeini et al. (2018)). This introduces the need to create computer models that can effectively combine and analyze different types of information. The task is complex and poses a big challenge in the field of NLP, as it requires the creation of advanced algorithms and models capable of understanding language almost like humans. Thus, the goal in detecting sarcasm goes beyond just improving how the text is analyzed; it is about developing systems that can fully understand and interpret the complex ways in which humans communicate.

As stated by Băroiu and Trăușan-Matu (2022), several studies have employed machine learning models such as logistic regression and support vector machines, alongside deep learning models like bidirectional long short-term memory networks and BERT (Bidirectional Encoder Representations from Transformers). These models have been applied to large datasets, including millions of social media comments, to refine the accuracy of sarcasm detection. Notably, deep learning models have shown promising results(Šandor and Babac (2023)). This underscores the advanced capabilities of deep learning in interpreting the nuanced aspects of language. Despite these advancements, there are a few limitations, especially concerning the nature of the datasets used. The existing research has mainly focused on social media content like twitter feeds (Afiyati et al. (2020)) which might contain noise in the form of informal language, slang, and varied contexts. This may lead to inaccuracies in sarcasm detection. This raises the need for exploring domain-specific dataset. For example, the news headlines dataset (Misra and Arora (2023)) has a distinct linguistic style which is different from the casual nature of social media language. Here, sarcasm is more structured and subtle, which presents a distinct set of challenges for sarcasm detection algorithm. By focusing on such datasets, the accuracy and adaptability of sarcasm detection models can be enhanced, making it more effective across various contexts.

The integration of deep learning architectures with this, can further advance the field. This raises the research question:**"How can the integration of diverse deep learning architectures, such as RNN, CNN, BiLSTM, and GRU, enhanced by the contextual capabilities of GloVe embeddings, improve the accuracy and robustness of sarcasm detection in specialized domains such as news headlines, compared to traditional machine learning models?"**

In this project, a hybrid deep learning model is implemented to address this research question. This model combines various deep learning architectures – Recurrent Neural Networks (RNN), Convolutional Neural Networks (CNN), Bidirectional Long Short-Term Memory Networks (BiLSTM), and Gated Recurrent Units (GRU) – each offering unique strengths in analyzing textual data. Traditional machine learning models are also built, and their performance is compared. This report is structured into five sections: 'Related Work' discusses existing studies in the field. 'Methodology' then explains the approaches used in the research. 'Design and Implementation' describes how the project was de-

veloped. The 'Evaluation' section will detail the testing and analysis of the model, assessing its performance in sarcasm detection. The report concludes with 'Conclusion and Future Work' summarizing the findings and suggesting areas for further research.

2 Related Work

2.1 Sophisticated Text Pre-processing Paradigms in NLP

Text pre-processing is a crucial aspect of Natural Language Processing that focuses on how machines understand and interpret human language. This is done by transforming the human language into a numerical format that is meaningful for computational interpretation. This is highly vital, especially in modern communication where slang and short forms are prevalent. A survey conducted by Tabassum and Patil (2020) highlights various pre-processing techniques and their relevance in making documents analyzable by machines. The paper enumerates several commonly used pre-processing techniques, such as sentence segmentation, lowercasing, tokenization, POS tagging, stopwords removal, punctuation removal, stemming, and lemmatization. The paper describes tokenization as the division of text into individual elements such as words and punctuation. It characterizes stopwords removal as the exclusion of frequently used words that do not contribute to the overall meaning such as "the" and "is." It further explains stemming as a process to truncate words to their root form and lemmatization as the reduction of words to their meaningful base, or lemmas. In addition to pre-processing, the paper explores feature extraction techniques like Named Entity Recognition (NER), Bag-of-Words (BoW), and TF-IDF. These are presented as crucial steps in transforming pre-processed text into vector forms, which are then utilized in classifier models. The paper emphasizes that not all techniques are universally suitable; their selection depends on the specific requirements of each use case.

Expanding on these concepts, a study by Vijayarani et al. (2015) explores several types of stop word removal methods, such as the Classic Method, Zipf's Law-based methods, the Mutual Information Method, and Term Based Random Sampling. In terms of stemming, it delves into different approaches like truncating methods, statistical methods, and mixed methods. Among the various methods discussed, the paper specifically highlights Porter's stemming algorithm which is renowned for its five-step process. Each step in this process methodically applies rules to strip suffixes from words. Significantly, it also references Porter's development of 'Snowball' framework which is designed to enable the creation of stemmers for different languages.

Building upon the foundational insights provided by the previous papers, Palomino and Aider (2022) extend the exploration of text pre-processing by focusing on its critical role in sentiment analysis, especially in social media contexts where unconventional text forms are common. Pre-processing normalizes the text and enhances the performance of sentiment classifiers. It specifically examines the effects of tokenization and lemmatization on the accuracy of sentiment analysis classifier such as naïve bayes. The paper concluded that the order of pre-processing steps significantly impacts the classifier performance.

Thus, pre-processing is a crucial step in enhancing the accuracy of sentiment analysis, making it an absolute necessity in the realm of Natural Language Processing. These studies, however, lack comprehensive pre-processing strategies that address the complex and varied nature of sarcasm in different text forms.

2.2 The Nuanced Dynamics of Word Embedding Mechanisms

Word embeddings are a form of word representations that links human understandings to machine understandings. These representations can be a set of real numbers or vectors. A study by Birunda and Devi (2021) has explored the various word embedding techniques for text classification. Three domains namely, traditional word embedding, static word embedding, and contextualized word embeddings were studied. The Glove is a static word embedding which is a count based unsupervised model. It combines the local context window method and global matrix factorization features. It predicts surrounding words by optimizing probability using a log bilinear model and weighted least squares. However, it requires a high memory for storage. Similarly, all the other embedding techniques such as TF-IDF, Word2Vec, Fast Text, BERT etc., were explored. This review concluded that these embeddings can be used with neural networks for text classification tasks, document clustering and sentiment analysis to improve model accuracy.

Incorporating insights from this, another study by Jiao and Zhang (2021), provides a comprehensive overview of the evolution of word embeddings in natural language processing. It is categorized into two methodologies namely, global matrix factorization methods like Latent Semantic Analysis (LSA) and sliding-window based methods such as Skip-Gram and CBOW. GloVe embeddings which combine the merits of LSA and CBOW are also explored. It is revealed that GloVe is training efficient and scalable to huge corpora. In addition to these, the recent advancements such as BERT, EMLO and GPT which are contextualized embeddings are studied. These methods help in overcoming the issue of polysemy where the semantics of a specific word may vary across different contexts.

Ni and Cao (2020) demonstrated the practical application of GloVe in sentiment analysis. This combination was chosen to overcome the limitations of traditional recurrent neural networks in learning long-term text information. The research findings show that the LSTM-GRU model, when used with GloVe embeddings, achieves high accuracy (87.10%) and F1 scores (86.76%) in sentiment analysis. This performance is comparable to the best outcomes achieved by LSTM models alone. Similarly, Zanchak et al. (2021) implemented neural network and machine learning models for sarcasm detection using various embeddings. Their findings indicate that the neural network model using GloVe embeddings achieved the highest accuracy at 80.5%. This was followed by the Bayesian classifier using TF-IDF vectors at 78.9%, and a neural network model with Word2Vec at 77.2%. This research emphasizes the effectiveness of embeddings in complex language tasks like sarcasm detection.

2.3 Exploring the Complexities of Sarcasm in Computational Linguistics

Sarcasm is a form of irony used to express opinions or feelings often opposite the literal meaning. There are significant challenges in sentiment analysis for detecting sarcasm, especially on social media platforms. Aboobaker and Ilavarasan (2020) have conducted a study to highlight the challenges in sarcasm detection and the various approaches in analyzing sentiments. The paper details the general architecture of sarcasm detection, which includes data collection (often from Twitter using hashtags like sarcasm), data pre-processing (such as tokenization and removal of stop words), feature extraction and selection (using methods like TF-IDF and N-grams), and classification techniques (like

SVM). The challenges identified include the difficulty in detecting sarcasm from text compared to speech, the ambiguity of sarcastic sentences, the complexity of sarcasm detection from noisy text due to limited context and features, and the need for additional features like semantic and text author-related elements, as sarcastic sentences often convey negative sentiments using positive words. These insights contribute to our understanding of sarcasm in computational linguistics and pave the way for future research aimed at enhancing the accuracy of sentiment analysis models.

2.4 Machine Learning Innovations in Textual Sentiment Analysis

With the evolution of Machine Learning techniques, the ability to analyze sentiment in texts has improved significantly. Pawar and Bhingarkar (2020) utilized a pattern-based approach with machine learning classifiers such as Support Vector Machine, Random Forest, and K-nearest Neighbor for detecting sarcasm in Twitter data. 9,104 tweets containing sarcasm, and not are used for this study. Post pre-processing and feature extraction models were built, and it is revealed that Random Forest obtained the highest accuracy of 81% and F1-Score of 79.00%. Expanding on this theme, Godara et al. (2022) collected a dataset of 34,000 tweets using the Tweepy API. After pre-processing and converting the unstructured data to a structured format, five models namely, Naïve Bayes, Decision Tree, AdaBoost, K-nearest Neighbor and Support Vector Machine classifiers were applied on the training and validation set. The test set was used to evaluate the performance of each classifier for detecting sarcastic text. Among the classifiers, Naïve Bayes had the highest accuracy of 61.18% and Decision Tree performed the worst with an accuracy of 54.27%.

Delving into a different dataset, Nayel et al. (2021) performed sentiment analysis on ArSarcasm-v2 dataset using Support Vector Machine. The performance of Support Vector Machine was compared with Linear Regression, Naïve Bayes, Complementary Naïve Bayes, and Stochastic Gradient Descent. Support Vector Machine outperformed all these models with an accuracy of 85%. Further exploring the field, Godara et al. (2021) introduced a scheme to detect sarcasm based on PCA (Principal Component Analysis) algorithm, K-Means algorithm, and ensemble classification. Four ensemble models with different combinations of Support Vector Machine, Logistic Regression, Decision Tree, K-nearest Neighbor, MLP were implemented, and their performance was tested on five datasets of varied sizes. It is revealed that the ensemble model in which Support Vector Machine, Logistic Regression and Decision Tree algorithms were combined had performed well for all five datasets as compared to the other combinations.

Each of these papers employ different machine learning techniques and datasets to tackle the challenge of sarcasm detection in textual sentiment analysis. The variation in performance metrics across these studies highlights the influence of dataset characteristics on the effectiveness of different ML models. Thus, the choice of the dataset plays a crucial role in determining the success of different models.

2.5 Advanced Deep Learning Approaches in Sarcasm Detection

Deep Learning, which is a subset of machine learning, has rapidly evolved by offering increasingly advanced architectures and algorithms that enhance the ability to interpret and analyze complex patterns in textual data. These advancements have been applied

in the nuanced task of sarcasm detection in many ways. One such implementation was done by Salim et al. (2020) where Recurrent Neural Network, Long Short-Term Memory (LSTM) and Word Embeddings were used for sarcasm detection. For this, a self-designed dataset of sarcastic and non-sarcastic tweets were utilized. The model achieved 85.23% sarcasm accuracy and 86.47% non-sarcasm accuracy with just 15 epochs, outperforming other algorithmic models.

Building on this, Jain et al. (2020), proposed a hybrid model of bidirectional long short-term memory with a SoftMax attention layer and convolution neural network. A bilingual dataset of 3000 sarcastic and 3000 non-sarcastic tweets were used. The model utilizes pre-trained GloVe word embeddings for extracting English semantic context vectors, a subjective lexicon Hindi-SentiWordNet to generate the HindiSenti feature vector, and additional pragmatic features indicating the count of pragmatic markers in tweets. The softAtt BiLSTM-feature-rich CNN (Convolutional Neural Network) model is trained using these feature vectors. The model achieved a classification accuracy of 92.71% and an F-measure of 89.05%, outperforming baseline deep learning models.

Kumar et al. (2020) utilized the SARC 1 corpus, a large self-annotated dataset containing over a million sarcastic and non-sarcastic comments from Reddit. The study involved building feature-rich Support Vector Machine (SVM) and Bidirectional Long Short-Term Memory (BiLSTM) models, including a variation of MHA-BiLSTM without auxiliary features. These models were used to classify the text as sarcastic or non-sarcastic. The results revealed that the multi-head attention mechanism enhances the performance of BiLSTM and also performs better than SVM. Transformer-based embeddings combined with Long Short-Term Memory Network was implemented by Nayak and Bolla (2022) for sarcasm detection on the News Headlines dataset. In this study, various vectorization techniques like TF-IDF, Word2Vec, Doc2Vec, and BERT were evaluated with seven classification algorithms. BERT with LSTM/Bi-LSTM yielded the best results, demonstrating high accuracy and performance metrics. Using the same dataset, Liu and Xie (2021) proposed a BERT-LSTM sarcasm detection model which achieved a high accuracy of 91.8%. This outperformed other models such as Word2Vec-LSTM and BERT-CNN. Savini and Caragea (2022) present a novel approach to sarcasm detection in natural language processing (NLP). They utilized three datasets of varying sizes and characteristics: the Sarcasm V2 Corpus, the Self-Annotated Reddit Corpus (SARC), and a Twitter dataset (SARCTwitter). The experiments conducted on all three datasets showed that the BERT model, with and without intermediate-task transfer learning, outperformed the state-of-the-art models, such as SVMs (Support Vector Machines) with Word2Vec and N-grams.

These studies collectively illustrate the dynamic and progressive nature of deep learning in sarcasm detection highlighting the potential of advanced methodologies to interpret the complexities of human language with increasing precision and effectiveness. While deep learning methods are increasingly used, there is a gap in studies that integrate multiple advanced architectures like RNN, CNN, BiLSTM, and GRU for a more robust approach to sarcasm detection in specialized domains.

2.6 The Multifaceted World of Topic Modeling

Topic modeling serves as a powerful tool in natural language processing, enabling the extraction and discovery of hidden thematic patterns within large collections of text, thereby simplifying the understanding and organization of complex data. Vayansky and

Kumar (2020) explored the various topic modelling methods that can deal with the correlation between topics, handling short texts and changes of topics over time. Approaches such as Latent Dirichlet Allocation (LDA), Correlated Topic Modeling (CTM), Pachinko Allocation Model and Dynamic Topic Modeling were explored. The study concluded that LDA is the most common method. Also, a decision tree was created by using each model's characteristics which will enable the users to choose the appropriate one for research. Extending on this, Egger and Yu (2022) evaluated the performance of four topic modelling techniques namely, LDA, non-negative matrix factorization (NMF), Top2Vec, and BERTopic. This research highlights the effectiveness of using BERTopic and Non-Negative Matrix Factorization (NMF) methods to analyze and understand Twitter data, focusing on specific analytical details and quality aspects.

Gregory et al. (2020) explored the use of Latent Dirichlet Allocation (LDA) topic models for sarcasm detection but found them ineffective. It was determined that the LDA topic models, even those with the best coherence, were not predictive of sarcasm. When a tweet's sarcasm was predicted based on its top three topics using either a Support Vector Machine (SVM) or Logistic Regression classifier, a training accuracy of only 53% was achieved. This study concluded that these features were unlikely to generalize well.

2.7 Future Directions and Emerging Trends in NLP for Sarcasm Detection

Băroiu and Trăușan-Matu (2022) outlined the evolution of sarcasm detection methods in NLP, transitioning from rule-based to deep learning models. A list of all datasets available and the model combinations implemented so far has been explored. The paper points out future research areas, highlighting the importance of using varied and complex datasets to understand sarcasm and investigate how context affects its detection, aiming to improve interactions between humans and machines more accurately. Rahma et al. (2023) presented a comprehensive study on Arabic Sarcasm Detection. This study highlighted the unique challenges and methodologies required for effective sarcasm analysis in languages with complex linguistic features. It focuses on the importance of feature extraction techniques, such as data cleaning, normalization, and tokenization, tailored to the Arabic language's specific characteristics. This aligns with future trends in NLP, where there is an increasing focus on developing models that can understand and interpret sarcasm not just in English but across diverse languages and cultures.

2.8 Conclusion

This review extensively explores text preprocessing, word embeddings, and sarcasm detection within NLP. It identifies key gaps such as the need for more holistic preprocessing techniques, the integration of diverse deep learning models, and the development of nuanced sarcasm detection methods tailored to specific contexts like news headlines.

3 Methodology

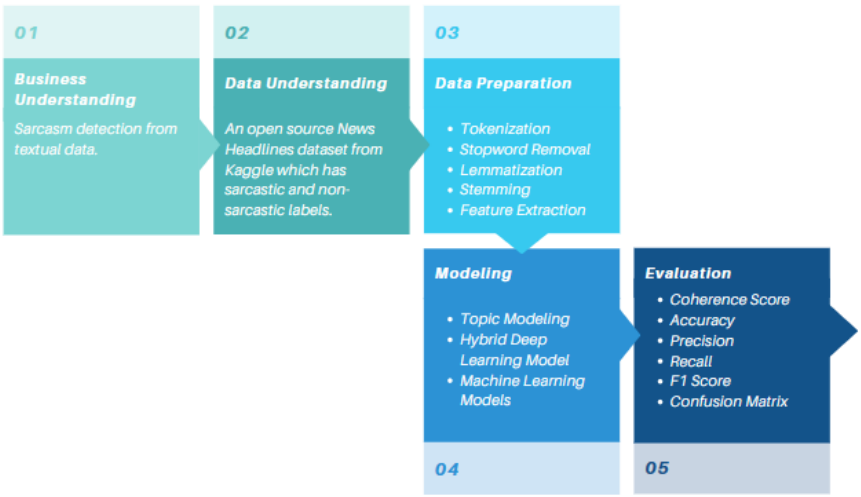

Figure 1: CRISP-DM Methodology

The methodology for Sarcasm Detection employs CRISP-DM (Schröer et al. (2021)), a widely recognized standard in the data mining field. This method is known for its clear and organized way of handling data projects. It helps in systematically approaching the challenge of identifying sarcasm in text data, from understanding the problem to applying the findings. A visual representation of the CRISP-DM process can be found in Figure 1

3.1 Business Understanding

In this initial phase, the project's objectives, and requirements, which is to identify sarcasm in textual data are defined. This requires a carefully crafted strategy to manage the complexities of language details and the interpretation of context, which are fundamental in recognizing sarcasm.

3.2 Data Understanding

The News Headlines dataset[1] that is used in this project is taken from Kaggle. The data is collected from two news websites namely The Onion and HuffPost. It contains 26,710 records and 3 columns – headline, is_sarcastic(binary column) and the article_link. The dataset is a JSON file which is imported to Jupyter Notebook for further analysis. This dataset has several advantages over the existing Twitter datasets. There are no spelling mistakes and informal usage as it is written by Professionals. Hence, the sparsity is reduced which increases the chance of finding pre-trained embeddings. Also, high quality labels with less noise are present. Thus, this makes the dataset well-suited for achieving the project's objectives.

[1] `https://www.kaggle.com/datasets/rmisra/news-headlines-dataset-for-sarcasm-detection`

3.3 Data Preparation

This phase is crucial as it involves cleaning and pre-processing the data, making it fit for model development. Once the dataset is loaded into Jupyter Notebook, it is converted to a Pandas Dataframe to facilitate efficient analysis and manipulation. Exploratory data analysis is done to gain more insights into the data. These include determining the distribution of sarcastic and non-sarcastic headlines in the form of a pie-chart and a bar chart for identifying the frequently occurring words. Pre-processing techniques such as tokenization, stopword removal, lemmatization and stemming are then done. Feature extraction methods, including vectorization, n-grams, and meta features, are also performed to enhance the dataset, making it more informative and relevant for the predictive models that follow. Libraries such as pandas, NumPy, nltk and matplotlib are used here. This comprehensive preparation lays the groundwork for effective model training and accurate sarcasm detection.

3.4 Modeling

Initially, an LDA (Latent Dirichlet Allocation) Model is constructed to identify the different topics present in the News Headlines. Utilizing the gensim library, a dictionary and a corpus are developed. Subsequently, seven distinct topics are identified and visualized for further analysis. Following this, the project advances to its crucial stage: the detection of sarcasm within the news headlines. For this task, two approaches are implemented – Deep Learning and Machine Learning, utilizing key libraries such as Keras, TensorFlow, and scikit-learn. For deep learning, a hybrid model which includes CNN (Convolutional Neural Network), RNN (with Bi-directional LSTM (Long Short-Term Memory) and GRU Layers) with Glove Embeddings is done. Once the dataset is split into training and test data, tokenization is done to convert the texts to a sequence of integers. Glove Embeddings are also generated. The hybrid model is then defined and trained using the training data. Post-training, it is applied to the testing data, and its performance is evaluated. For the Machine Learning approach, two algorithms are chosen - Random Forest and Decision Tree. These models are subjected to hyperparameter tuning through grid search to identify the most effective parameters, ensuring optimal performance.

3.5 Evaluation

The LDA model is evaluated using the coherence score metric. The classification models are evaluated using metrics like accuracy, precision, recall, F1 score and a confusion matrix (Vujović (2021)). Based on these metrics, the best approach is identified.

4 Design Specification and Implementation

The below Figure 2 represents the high-level architecture design. The news headlines dataset which is in JSON format is taken from Kaggle and loaded into Jupyter Notebook. It is then converted into a Pandas Dataframe post which exploratory data analysis and pre-processing is done. Topic modeling is implemented using LDA and is evaluated using the coherence score metric. For sarcasm detection, two approaches are implemented. A hybrid deep learning model using Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN) with Bidirectional Long Short-Term Memory (BiLSTM)

and Gated Recurrent Units (GRU), alongside GloVe embeddings is built and evaluated. Similarly, two traditional machine learning models namely, Random Forest and Decision Tree are also implemented and evaluated. The best model is then chosen based on the performance metrics.

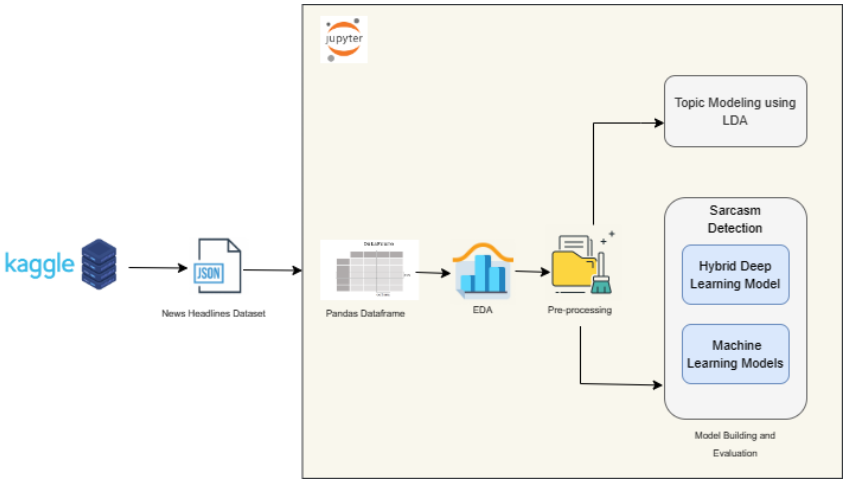

Figure 2: High-Level Architecture

4.1 Data Preparation Phase

4.1.1 Data Loading and Visualization

The News Headlines Dataset is taken from Kaggle and contains 27,170 records and 3 columns – article_link, headline and is_sarcastic. The is_sarcastic column is binary and contains values 0 or 1. 0 is for non-sarcastic and 1 is for sarcastic. The dataset is in JSON format. This data is loaded into Jupyter Notebook and converted to a Pandas Dataframe. A snippet of the data is shown in the below Figure 3.

	article_link	headline	is_sarcastic
0	https://www.huffingtonpost.com/entry/versace-b...	former versace store clerk sues over secret 'b...	0
1	https://www.huffingtonpost.com/entry/roseanne-...	the 'roseanne' revival catches up to our thorn...	0
2	https://local.theonion.com/mom-starting-to-fea...	mom starting to fear son's web series closest ...	1
3	https://politics.theonion.com/boehner-just-wan...	boehner just wants wife to listen, not come up...	1
4	https://www.huffingtonpost.com/entry/jk-rowlin...	j.k. rowling wishes snape happy birthday in th...	0

Figure 3: Sample Data

Exploratory Data Analysis (EDA) is conducted to delve deeper into the dataset. A pie chart (as shown in Figure 4) is used to visualize the distribution of sarcastic versus non-sarcastic labels, revealing a balanced representation of both categories. Additionally, a bar chart is created to display the most commonly occurring words in the headlines. The presence of numerous common words, as indicated by this chart, underscores the necessity for thorough pre-processing of the data.

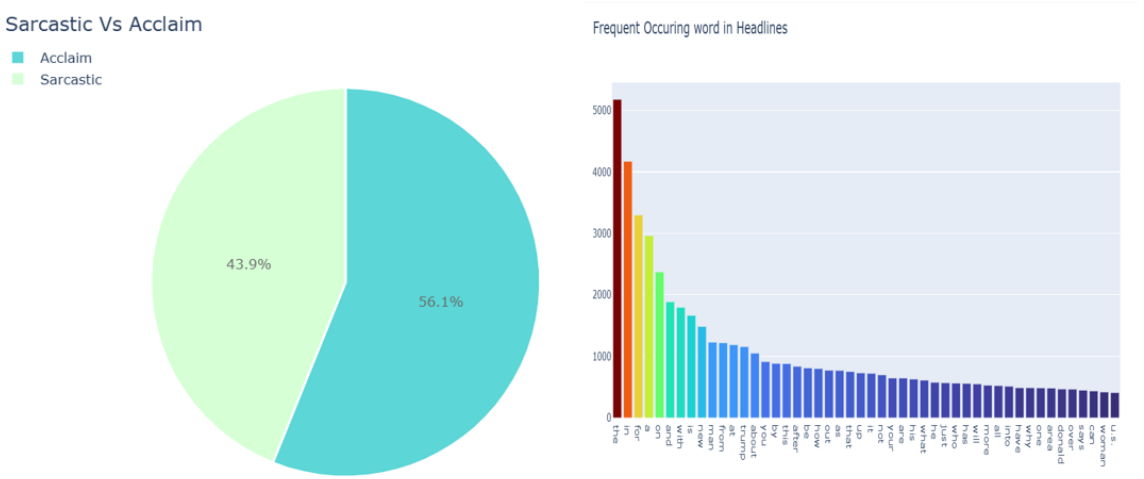

Figure 4: Distribution of Labels and Frequently Occurring Words

4.1.2 Data Pre-processing and Feature Extraction

In this phase of the project, a meticulous pre-processing and feature extraction process is conducted on the dataset to prepare it for effective sarcasm detection. This phase is crucial as it transforms raw text data into a structured format that can be efficiently analyzed and processed by machine learning models. The pre-processing steps are as follows. These steps are vital for reducing noise in the data and focusing the analysis on the words most relevant to sarcasm detection.

i. Tokenization: The headlines are tokenized, breaking down text into individual words or tokens. This step is important for understanding the basic units of meaning within the text.

ii. Stopword Removal: Commonly used words (stopwords) that offer little value in the context of analysis are removed. This includes words like 'the', 'is', and 'in'. After removing the stop words, the length of the sarcasm and acclaim list reduces from 263026 words to 190983 words.

iii. Lemmatization: Words are reduced to their base or root form, helping in the standardization of the text for analysis.

iv. Stemming: This process further reduces words to their stem form, stripping suffixes to consolidate similar words into one token.

After pre-processing, the below feature extraction techniques are applied.

i. Vectorization: The tokenized words are converted into numerical values using vectorization techniques, making them interpretable by machine learning algorithms.

ii. N-Grams Creation: N-grams (bigrams and trigrams) are generated to capture the context by considering the sequence of words. This helps in understanding the phrase structure, which is crucial in sarcasm detection.

iii. Additional Text Features: Various text attributes like the number of words, unique words, characters, stopwords, punctuations, and the average word length are calculated. These features provide further insights into the structure and complexity of the text.

These steps enhance the quality and usability of the textual data for modeling. By focusing on the relevant features, the efficiency of the machine learning models can be improved.

4.2 Modeling Phase

4.2.1 Topic Modeling using LDA

Topic Modeling is done on the 'headlines' column to identify the different topics present. The data is first converted into a list format, which is a necessary step for processing data in topic modeling. A dictionary is then created which serves as a mapping between words in the data and their unique integer IDs. These indexes the words in the dataset. A document-term matrix (corpus) is generated using the dictionary. This matrix represents the frequency of each word (from the dictionary) in each document (headline) in a bag-of-words format. The LDA model is then constructed with the corpus and dictionary as inputs. The model is configured to discover seven different topics. Other parameters such as random_state, chunksize, passes, and per_word_topics are also set. The model outputs the top words for each topic, which are printed to understand the thematic composition of each topic. A function is then created to visualize each of the seven topics using word clouds, which graphically represent the most frequent words in each topic. The following Figure 5 illustrates the inner workings of the LDA algorithm (Buenano-Fernandez et al. (2020)).

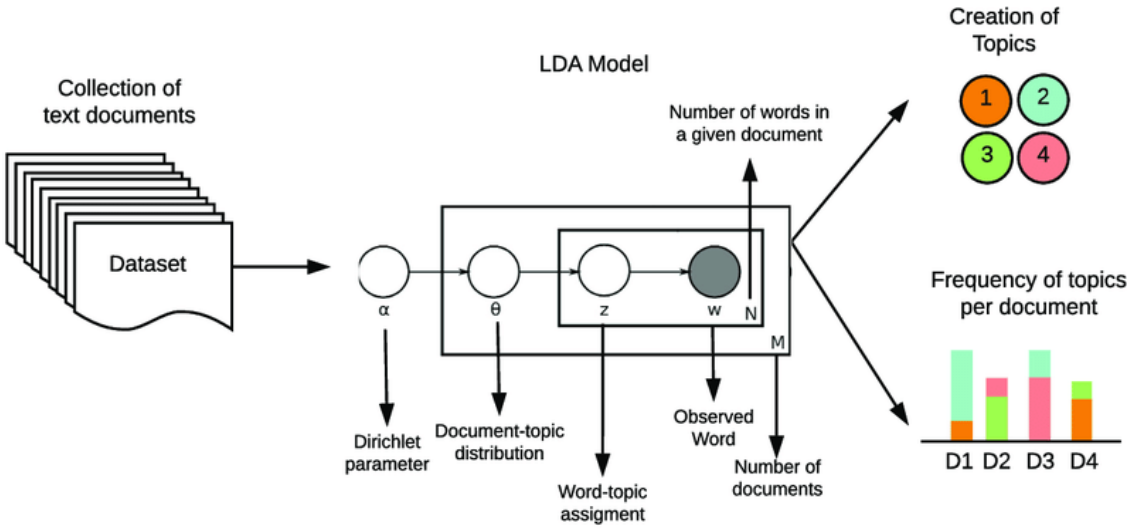

Figure 5: Working of LDA

4.2.2 Hybrid Deep Learning Model for Sarcasm Detection

The hybrid model for sarcasm detection is a complex combination of CNN, RNN with Bidirectional LSTM and GRU layers where GloVe embeddings are provided as input. Sev-

eral foundational steps are implemented to prepare the dataset for modeling. Headlines column is the independent variable and is_sarcastic column is the dependant variable. The dataset is divided into training and testing using a standard train-test split with 20% of the data reserved for testing. This helps in validating the model's performance on unseen data. Next, tokenisation is performed on the training data where the headlines are converted to a sequence of integers. Here, each integer represents a specific word in the dataset. In addition to this, pre-trained GloVe (Global Vectors for Word Representation) embeddings are utilized to provide a rich, pre-established understanding of word contexts and relationships. By doing so, the words are mapped to high-dimensional vectors which helps in capturing the semantic meanings based on co-occurrence in a large corpus. The embedding matrix is then adapted to the dataset's vocabulary. This ensures that each word in the dataset is represented by its corresponding GloVe vector. The hybrid model is then constructed as shown in the below Figure 6.

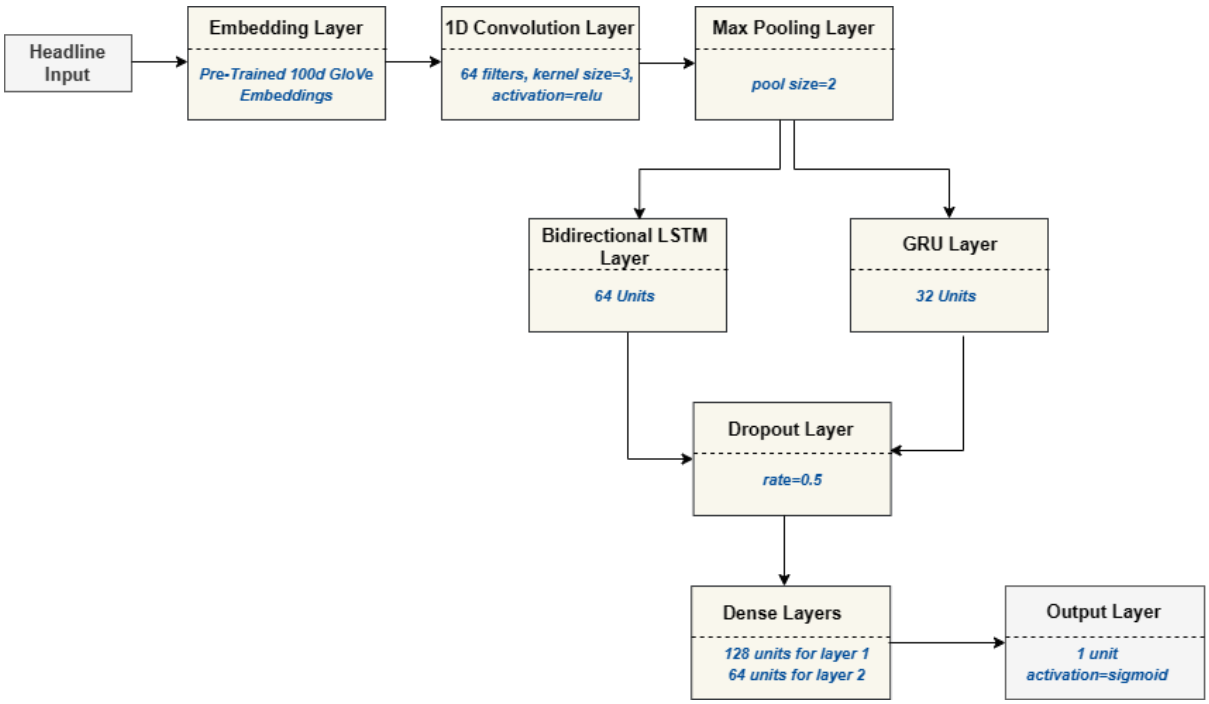

Figure 6: Hybrid Deep Learning Model Architecture

i. Input and Embedding Layer

- The model starts with an input layer which receives sequences of tokenized and padded text data with a maximum length of 150.
- This is followed by an embedding layer which uses the pre-trained GloVe embeddings with 100-dimensional word vectors. This provides a dense and meaningful representation of words.

ii. Convolutional Neural Network (CNN)

- Next comes the CNN layer which consists of 64 filters and a kernel size of 3. This layer extracts the local and temporal features from the text which will aid in capturing the patterns indicative of sarcasm.

- A MaxPooling layer is then defined which has a pool size of 2. This reduces dimensionality, emphasizing the most relevant features.

iii. Bidirectional LSTM

- The long-term dependencies are then captured by the 64 units LSTM layer. Here, the data is processed in both forward and backward directions so that a comprehensive context understanding is achieved.

iv. Gated Recurrent Unit (GRU)

- A GRU layer with 32 units effectively manages the flow of information, adapting to different time scales of dependencies.

v. Regularization and Fully Connected Layers

- Dropout layers with a rate of 0.5 are defined. This helps in preventing overfitting.
- Two fully connected dense layers with 128 and 64 units respectively are then defined. This includes L1/L2 regularization. It introduces non-linearity and enhances the learning capability for classification.

vi. Output Layer

- Finally, a dense layer with a sigmoid activation function outputs the probability of sarcasm.

After the model configuration, it undergoes training using the training dataset with 10 and 20 epochs, with the learning rate set at 0.0001 and a batch size of 100.

4.2.3 Machine Learning Models for Sarcasm Detection

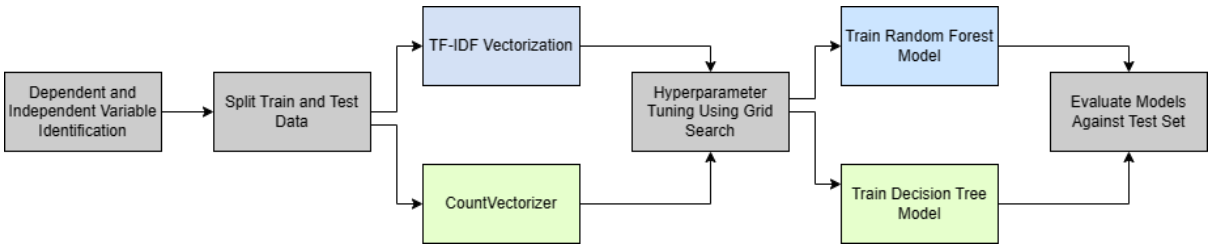

Figure 7: Decision Tree and Random Forest Flow

Following the neural network approach, two established machine learning algorithms, Random Forest and Decision Tree Classifiers, are also employed (refer Figure 7). The dataset is first prepared by separating the 'Headline' text, which serves as the independent variable, from the 'is_sarcastic' labels, which form the target variable. For Random Forest, the text data is converted into a numerical format using TF_IDF (Term Frequency-Inverse Document Frequency) Vectorizer. This vectorizer transforms the 'Headline' text data into a matrix of TF-IDF features. This helps in capturing the importance of words relative to the document and the entire corpus. The transformed data is then split into training and

testing sets with 20% reserved for evaluation. A grid search technique is implemented for hyperparameter tuning to determine the most effective parameters for the Random Forest Classifier. After tuning, the classifier is configured with 200 estimators and uses the 'gini' criterion for the quality of splits, with min_samples_split and min_samples_leaf set to 2 and 1, respectively. For the decision tree, a similar 80-20 split is done, post which the text data is vectorized using the CountVectorizer. This is configured to capture unigrams. Hyperparameter tuning is carried out through which the optimized values are determined. The max_depth is set to 10, while min_samples_split and min_samples_leaf is set to 5 and 4, respectively. Both the models are then trained on the dataset using these refined parameters, readying it for evaluation against the test set.

5 Evaluation

5.1 Experiment 1: LDA Evaluation

The LDA model is evaluated using the coherence score metric(O'callaghan et al. (2015)). Below are the topics (refer Figure 8) returned by the model and the score obtained was 0.4418.

```
Topic: 0
Words: 0.026*"american" + 0.025*"report" + 0.021*"woman" + 0.020*"area" + 0.016*"america" + 0.012*"could" + 0.011*"ban"
Topic: 1
Words: 0.052*"man" + 0.019*"obama" + 0.016*"day" + 0.015*"first" + 0.015*"still" + 0.011*"ever" + 0.010*"clinton"
Topic: 2
Words: 0.042*"trump" + 0.012*"donald" + 0.011*"like" + 0.010*"house" + 0.010*"women" + 0.010*"best" + 0.008*"white"
Topic: 3
Words: 0.022*"year" + 0.021*"one" + 0.019*"says" + 0.015*"old" + 0.015*"make" + 0.011*"fall" + 0.010*"tips"
Topic: 4
Words: 0.017*"got" + 0.015*"2" + 0.015*"mom" + 0.014*"ryan" + 0.014*"free" + 0.013*"2014" + 0.010*"bad"
Topic: 5
Words: 0.012*"get" + 0.012*"world" + 0.011*"school" + 0.009*"pope" + 0.009*"look" + 0.009*"3" + 0.008*"francis"
Topic: 6
Words: 0.047*"new" + 0.014*"people" + 0.014*"time" + 0.012*"life" + 0.012*"nation" + 0.011*"back" + 0.011*"5"
```

Figure 8: LDA Topic Modeling Result

The topics obtained were then visualised using a WordCloud. This can be seen in the below Figure 9.

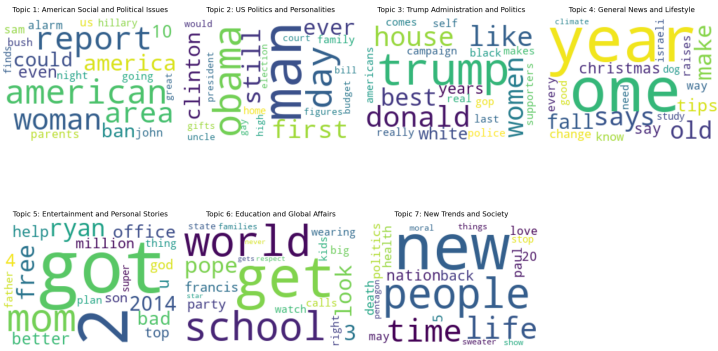

Figure 9: LDA Topic Modeling WordCloud

5.2 Experiment 2: Hybrid Deep Learning Model with 10 Epochs and 20 Epochs

The hybrid deep learning model which includes CNN, RNN(With Bidirectional LSTM and GRU Layers) and GloVe Embeddings is trained using 10 and 20 epochs. The early stopping mechanism is implemented to prevent overfitting. It ends the training process when the model stops improving on the validation set, thereby saving time and computational resources, and most importantly, preventing the model from learning the training data so well that it performs poorly on new, unseen data. restore_best_weights parameter is set to True. This means that, after early stopping is triggered, it restores the model weights from the epoch with the best value of the monitored quantity. The training and validation accuracy obtained after 10 epochs was 85.85% and 81.98% respectively. The training and validation loss scored was 0.3592 and 0.4063 respectively.

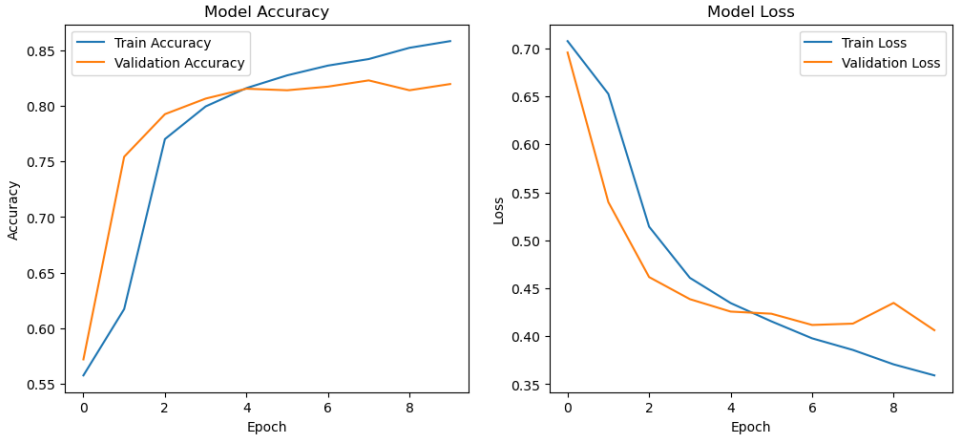

Figure 10: Hybrid Deep Learning Model Accuracy and Loss for 10 Epochs

After extending the training to 20 epochs, the hybrid deep learning model, demonstrated further improvements. With the early stopping mechanism activated, the training ceased at epoch 16 when no further improvement in validation loss was observed, ensuring the model did not overfit. The best weights were restored from epoch 11, which had the lowest recorded validation loss. At this point, the model achieved a training accuracy of 86.49% and a validation accuracy of 82.03%, with corresponding losses of 0.3444 for training and 0.4151 for validation.

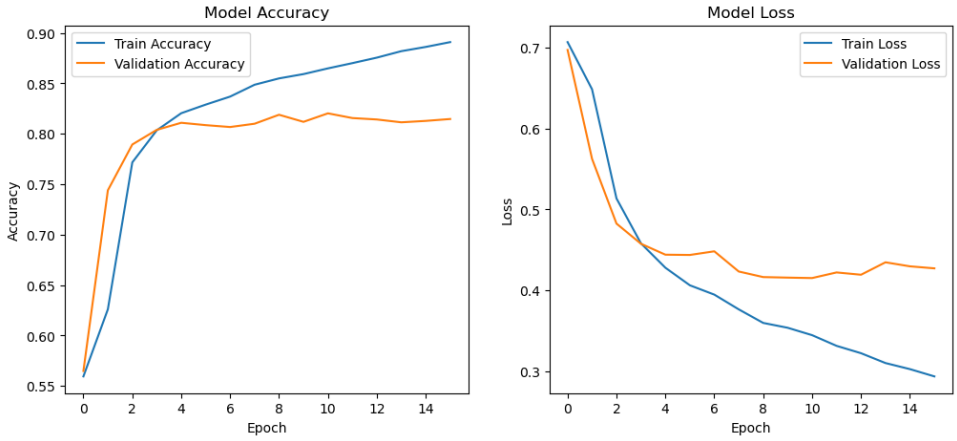

Figure 11: Hybrid Deep Learning Model Accuracy and Loss for 20 Epochs

5.3 Experiment 3: Testing the Hybrid Deep Learning Model on the Test Data

From the training sessions, it was evident that extending the hybrid deep learning model to 10 epochs resulted in a better performance as compared to 20 epochs. Using this, the test data predictions were carried out. Accuracy, Precision, F1-Score, recall metrics were calculated. In addition to this, a confusion matrix was also plotted. The results obtained are depicted in the Figure 12 below.

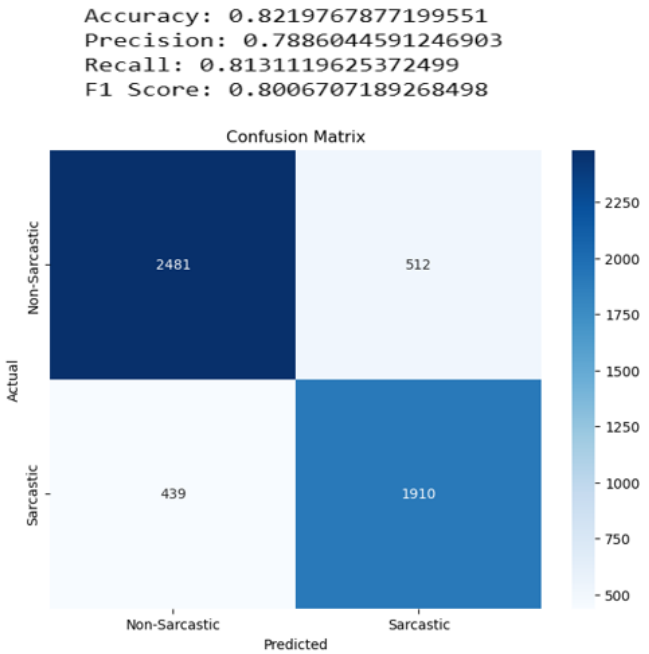

Figure 12: Hybrid Deep Learning Model Evaluation for Test Data

5.4 Experiment 4: Random Forest and Decision Tree Evaluation

Similar to the deep learning model, the performance metrics such as Accuracy, Precision, F1-Score, recall metrics and confusion matrix were calculated for the machine learning models as well. The results obtained are depicted in the Figure 13 below.

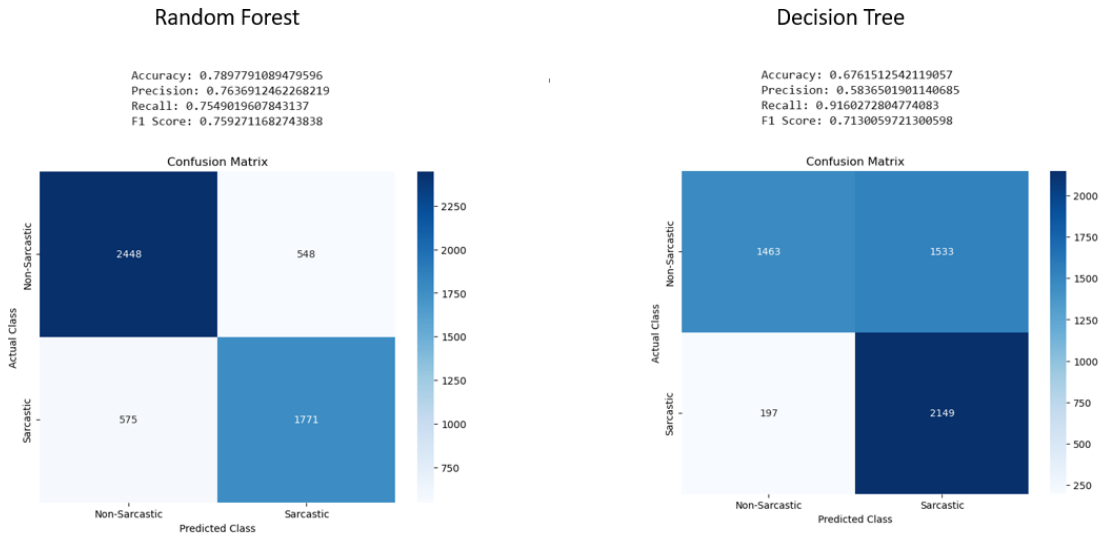

Figure 13: Machine Learning Model Evaluation for Test Data

5.5 Discussion

The evaluation shows that topic modeling resulted in a low coherence score of 0.4418. This suggests that the topics generated by the LDA model is not as coherent or meaningful as desired. A low coherence score often means that the topics contain words that do not seem to be related to each other very well, making it difficult to interpret what each topic is about. Thus, this model was not further used for sarcasm detection. This is similar to what had been concluded by Gregory et al. (2020). If a high coherence score from LDA was obtained, it would mean better topic categorization, which in turn could provide valuable context for more advanced sarcasm detection processes. However, it's important to note that LDA alone wouldn't detect sarcasm; it would simply enhance the performance of sarcasm detection algorithms by providing them with relevant contextual information. The second and third experiment done for sarcasm detection revealed that the hybrid deep learning model with 10 epochs resulted in a high training and validation accuracy of 85.85% and 81.98%. The training and validation loss was 0.3592 and 0.4063 respectively. This model was further tested on the test data and an accuracy of 82.19% was achieved. Experiment four done using random forest and decision tree resulted in an accuracy of 78.97% and 67.61% respectively. Given these outcomes, the hybrid deep learning model emerged as the superior choice for effective sarcasm detection.

6 Conclusion and Future Work

The main objective of this study was to explore how the integration of diverse deep learning architectures would improve the accuracy of sarcasm detection as compared to the traditional machine learning models, especially in the domain of news headlines. This was to be achieved by evaluating the effectiveness of the integrated models in detecting sarcasm more accurately and comparing their performance with that of traditional models. This involved implementing a hybrid deep learning framework that combined the strengths of RNN, CNN, BiLSTM, and GRU architectures. GloVe embeddings were utilized to capture contextual word relationships, providing a rich semantic understanding which is highly vital for sarcasm detection. The study successfully demonstrated that the hybrid model outperformed traditional machine learning approaches. This shows that the hybrid model is capable of understanding complex linguistic constructs like sarcasm. The integration of GloVe embeddings also contributed to it. This research is important because it can help improve computer systems that automatically check and manage online content. It can make these systems better at understanding emotions in text, which is especially useful for social media companies that want to make sure conversations on their platforms are clear and respectful. The study's success mainly depends on the availability of detailed and relevant data, and there is a risk that the model might learn the training data too well, leading to less accuracy with new, unseen data. It is also challenging to apply the results more broadly to different styles and nuances of language. For future research, integrating transformer models like BERT or GPT could be explored to leverage their self-attention mechanisms for even deeper analysis. Improving topic modeling techniques and their incorporation into the sarcasm detection process could refine the model's sensitivity to subject-specific sarcasm. A potential follow-up research project could explore the application of these models in different languages and cultural contexts to enhance the universality of sarcasm detection tools.

References

Aboobaker, J. and Ilavarasan, E. (2020). A survey on sarcasm detection and challenges, *Proc. of 6th Intl. Conf. on Advanced Computing & Communication Systems*, pp. 1234–1240.

Afiyati, A., Azhari, A., Sari, A. and Karim, A. (2020). Challenges of sarcasm detection for social network: a literature review, *JUITA: Jurnal Informatika* **8**(2): 169–178.

Birunda, S. S. and Devi, R. K. (2021). A review on word embedding techniques for text classification, *Innovative Data Communication Technologies and Application: Proceedings of ICIDCA 2020*, pp. 267–281.

Buenano-Fernandez, D., Gonzalez, M., Gil, D. and Luján-Mora, S. (2020). Text mining of open-ended questions in self-assessment of university teachers: An lda topic modeling approach, *IEEE Access* **8**: 35318–35330.

Băroiu, A. and Trăușan-Matu, (2022). Automatic sarcasm detection: Systematic literature review, *Information* **13**(8): 399.

Egger, R. and Yu, J. (2022). A topic modeling comparison between lda, nmf, top2vec, and bertopic to demystify twitter posts, *Frontiers in sociology* **7**: 886498.

Ghaeini, R., Fern, X. and Tadepalli, P. (2018). Attentional multi-reading sarcasm detection, *arXiv preprint arXiv:1809.03051* .

Godara, J., Aron, R. and Shabaz, M. (2022). Sentiment analysis and sarcasm detection from social network to train health-care professionals, *World Journal of Engineering* **19**(1): 124–133.

Godara, J., Batra, I., Aron, R. and Shabaz, M. (2021). Ensemble classification approach for sarcasm detection, *Behavioural Neurology* .

Gregory, H., Li, S., Mohammadi, P., Tarn, N., Draelos, R. and Rudin, C. (2020). A transformer approach to contextual sarcasm detection in twitter, *Proceedings of the second workshop on figurative language processing*, pp. 270–275.

Gupta, R., Kumar, J. and Agrawal, H. (2020). A statistical approach for sarcasm detection using twitter data, *2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS)*, IEEE, pp. 633–638.

Jain, D., Kumar, A. and Garg, G. (2020). Sarcasm detection in mash-up language using soft-attention based bi-directional lstm and feature-rich cnn, *Applied Soft Computing* **91**: 106198.

Jiao, Q. and Zhang, S. (2021). A brief survey of word embedding and its recent development, *2021 IEEE 5th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, Vol. 5, pp. 1697–1701.

Kumar, A., Narapareddy, V., Srikanth, V., Malapati, A. and Neti, L. (2020). Sarcasm detection using multi-head attention based bidirectional lstm, *Ieee Access* **8**: 6388–6397.

Liu, H. and Xie, L. (2021). Research on sarcasm detection of news headlines based on bert-lstm, *2021 IEEE international conference on emergency science and information technology (ICESIT)*, pp. 89–92.

Misra, R. and Arora, P. (2023). Sarcasm detection using news headlines dataset, *AI Open* **4**: 13–18.

Mukherjee, S. and Bala, P. (2017). Detecting sarcasm in customer tweets: an nlp based approach, *Industrial Management Data Systems* **117**(6): 1109–1126.

Nayak, D. and Bolla, B. (2022). Efficient deep learning methods for sarcasm detection of news headlines, *Machine Learning and Autonomous Systems: Proceedings of ICMLAS 2021*, Springer Nature Singapore, pp. 371–382.

Nayel, H., Amer, E., Allam, A. and Abdallah, H. (2021). Machine learning-based model for sentiment and sarcasm detection, *Proceedings of the Sixth Arabic Natural Language Processing Workshop*, pp. 386–389.

Ni, R. and Cao, H. (2020). Sentiment analysis based on glove and lstm-gru, *2020 39th Chinese control conference (CCC)*, pp. 7492–7497.

O'callaghan, D., Greene, D., Carthy, J. and Cunningham, P. (2015). An analysis of the coherence of descriptors in topic modeling, *Expert Systems with Applications* **42**(13): 5645–5657.

Palomino, M. and Aider, F. (2022). Evaluating the effectiveness of text pre-processing in sentiment analysis, *Applied Sciences* **12**(17): 8765.

Pawar, N. and Bhingarkar, S. (2020). Machine learning based sarcasm detection on twitter data, *2020 5th international conference on communication and electronics systems (ICCES)*, pp. 957–961.

Potamias, R., Siolas, G. and Stafylopatis, A. (2020). A transformer-based approach to irony and sarcasm detection, *Neural Computing and Applications* **32**: 17309–17320.

Rahma, A., Azab, S. and Mohammed, A. (2023). A comprehensive review on arabic sarcasm detection: Approaches, challenges and future trends, *IEEE Access* .

Salim, S., Ghanshyam, A., Ashok, D., Mazahir, D. and Thakare, B. (2020). Deep lstm-rnn with word embedding for sarcasm detection on twitter, *2020 international conference for emerging technology (INCET)*, pp. 1–4.

Savini, E. and Caragea, C. (2022). Intermediate-task transfer learning with bert for sarcasm detection, *Mathematics* **10**(5): 844.

Schröer, C., Kruse, F. and Gómez, J. (2021). A systematic literature review on applying crisp-dm process model, *Procedia Computer Science* **181**: 526–534.

Tabassum, A. and Patil, R. (2020). A survey on text pre-processing & feature extraction techniques in natural language processing, *International Research Journal of Engineering and Technology (IRJET)* **7**(06): 4864–4867.

Vayansky, I. and Kumar, S. (2020). A review of topic modeling methods, *Information Systems* **94**: 101582.

Vijayarani, S., Ilamathi, M. and Nithya, M. (2015). Preprocessing techniques for text mining-an overview, *International Journal of Computer Science & Communication Networks* **5**(1): 7–16.

Vujović, (2021). Classification model evaluation metrics, *International Journal of Advanced Computer Science and Applications* **12**(6): 599–606.

Zanchak, M., Vysotska, V. and Albota, S. (2021). The sarcasm detection in news headlines based on machine learning technology, *2021 IEEE 16th International Conference on Computer Sciences and Information Technologies (CSIT)*, Vol. 1, pp. 131–137.

Šandor, D. and Babac, M. (2023). Sarcasm detection in online comments using machine learning, *Information Discovery and Delivery* (ahead-of-print).