Deep Sarcasm: A Neural Approach to Detecting Contextual Irony in Online Communication

J Himabindu[1], J Swami Naik[2], B Dilip Kumar Reddy[3]
[1]*Student, G. Pulla Reddy Engineering College (Autonomous)*
[2]*Associate Professor, G. Pulla Reddy Engineering College (Autonomous)*
[3]*Assistant Professor, G. Pulla Reddy Engineering College (Autonomous)*

***Abstract*—Sarcasm is a complex form of expression where the intended meaning often contrasts with the actual words, usually to criticize or mock. On social media, such language makes automated interpretation especially difficult for tasks like sentiment analysis, opinion mining, and fake news detection. Traditional NLP techniques often fail to capture sarcasm, particularly when it involves slang, abbreviations, or context-dependent cues. In this paper, we propose a hybrid neural framework that integrates three embedding models Word2Vec, GloVe, and the lightweight DistilBERT. A fuzzy logic layer is introduced as the final decision-maker, balancing the outputs of these models to improve classification performance. Unlike earlier systems that depend on a single representation, our approach combines efficiency with stronger contextual understanding. The model was evaluated on three benchmark datasets: the Riloff Twitter Sarcasm dataset, the Reddit Comments Corpus, and a Kaggle News Headlines dataset. Experimental results show accuracies of 87.40%, 83.25%, and 91.10% across these datasets, confirming that the proposed method outperforms several existing approaches. Overall, the framework demonstrates strong potential for real-time and resource-constrained applications in sarcasm detection.**

***Index Terms*—BERT, FuzzyLogic, GloVe, Social Media.**

I. INTRODUCTION

With the rapid rise of social media, people now share opinions, emotions, and humor instantly with a global audience. Platforms such as Twitter and Facebook have become spaces not only for discussion but also for satire, criticism, and sarcasm. Sarcasm in text is especially tricky, as what may appear humorous to one reader could be offensive to another. Unlike spoken conversations, where tone and body language help reveal intent, online text lacks these cues, making detection much harder. Recent studies show that sarcasm plays an important role in shaping online communication. During the COVID-19 pandemic, for example, researchers observed that people experiencing stress, anxiety, or depression used sarcasm more frequently in their posts. Beyond entertainment, sarcasm creates challenges in several NLP tasks. Sentiment analysis,
fake news detection, cyberbullying identification, and opinion mining can all produce misleading results when sarcasm is misclassified. An apparently "positive" phrase may, in reality, express a negative or mocking intent. For these reasons, sarcasm detection has become an essential area of research in NLP. A reliable detection system must be capable of interpreting subtle cues, contextual contrasts, and hidden meanings in short pieces of text such as tweets, headlines, or memes. The work presented in the paper aims to address these challenges by introducing a hybrid framework that combines traditional embeddings with transformer-based models, supported by a fuzzy logic layer for better decision-making.

A. Challenges in Sarcasm Detection

The explosion of user-generated content on social media has created valuable opportunities for organizations and researchers to better understand public sentiment, behaviour and opinions. However, this wealth of data also brings unique challenges particularly when it comes to interpreting sarcasm. Sarcasm often uses language that appears positive on the surface, yet conveys a negative or mocking tone when placed in context. These subtleties can seriously mislead automated systems, resulting in inaccurate sentiment classification, flawed product review analysis, or even misidentification of false news. As the demand for deeper insights from textual

data grows, both academia and industry have become increasingly focused on improving sarcasm detection. While many natural language processing (NLP) models incorporate contextual information to interpret meaning, no single method can effectively handle the full spectrum of sarcastic content. This is partly due to differences in how various models interpret word proximity, sentence structure, and temporal context during training. Sarcasm is highly context-dependent, and different models often prioritize different types of relationships within the text.

B. Major Contributions

Researchers have explored various approaches to sarcasm detection by focusing either on content-based features (e.g., word patterns, syntax) or context-based signals (e.g., surrounding text, discourse history). However, these standalone strategies often fail to capture the full nuance of sarcasm, especially on fast-paced, diverse platforms like Twitter and Reddit. To address these challenges more productively, we propose a hybrid ensemble approach that combines the strengths of both content- and context-based methods. Our proposed model integrates three embedding techniques: Word2Vec**,** GloVe **and** DistilBERT (replacing BERT for improved efficiency). Word2Vec and GloVe represent word-level semantics based on neighborhood statistics, while DistilBERT a distilled version of BERT captures deeper, bidirectional sentence-level context. Each component independently generates a classification probability for whether a given input is sarcastic. These outputs are then passed to a fuzzy logic module, which serves as the final decision-maker. Instead of a rigid voting mechanism, fuzzy logic allows us to weigh the confidence of each model categorizing their outputs as high*,* medium*, or* low and apply rule-based reasoning to reach a final classification. This fusion strategy helps balance the individual limitations of each model, offering a more robust and interpretable solution. We validated our approach using three publicly available datasets: a Twitter sarcasm corpus, the Self-Annotated Reddit Corpus (SARC), and a headline dataset containing both factual and satirical titles. The model achieved strong results with accuracy scores of 87.40%, 83.25%, and 91.10%**,** respectively surpassing several existing state-of-the-art methods.

- We propose an efficient and scalable hybrid ensemble model for sarcasm detection in online text.
- The model uniquely combines word-level and sentence-level embeddings (Word2Vec, GloVe, DistilBERT) with a fuzzy logic controller for final classification.
- It was analysed on diverse, real-world datasets from Twitter, Reddit, and news headlines to ensure generalizability.
- The framework is designed to be adaptable for deployment across a variety of social media and e-commerce platforms.

This model has wide-ranging applications for both industry and research. Companies can use it to more accurately interpret user feedback, avoiding misclassification of sarcastic praise as genuine approval. Political organizations can gauge public sentiment more accurately by filtering out sarcasm from social media commentary. Likewise, fact-checking and misinformation detection systems can avoid false positives by accounting for ironic or sarcastic content. Furthermore, businesses can benefit from revisiting employee reviews and customer feedback that may contain sarcasm disguised as positive sentiment. Even public opinion surveys and social polls can yield more reliable insights when sarcastic noise is filtered out—leading to better decision-making at both organizational and societal levels.

II. RELATED WORK

Even though sarcasm has often come up in social science discussions, building systems to spot it in text is still a work in progress. In recent years, researchers in natural language processing (NLP) and machine learning (ML) have increasingly focused on this problem, particularly in connection with sentiment classification tasks. Early approaches mostly depended on linguistic features such as word frequencies, sentence patterns, and part-of-speech tags, combined with supervised learning techniques. For instance, Khodak et al. [8] introduced a large sarcasm dataset and compared manual annotations with automated methods such as vectorization and n-grams, showing that human labeling was still more reliable. Similarly, Eke et al. [9] reviewed existing

methods and noted that n-gram features, POS tagging, and chi-squared tests were among the most common tools, while classifiers such as Naïve Bayes, Random Forests, and SVMs were widely used. Sarsam et al. [10] further compared customized and adapted ML algorithms, finding that hybrid CNN-SVM models worked better when both lexical and user-specific features were considered. Traditional ML approaches mainly relied on handcrafted features. Keerthi Kumar and Harish [11], for example, used content-based features with feature selection methods like Information Gain and Mutual Information before applying clustering and SVM. Pawar and Bhingarkar [12] expanded this by adding recurring themes, punctuation, and emotion cues, and trained Random Forest and SVM models on these features. With the rise of deep learning, research moved beyond shallow features to capture sentence-level meaning. Ghosh and Veale [13] combined CNNs with bidirectional LSTMs for sarcasm detection on Twitter data, while Ghosh, Fabbri, and Muresan [14] extended this approach by incorporating conversational context from preceding tweets. Liu et al. [15] focused on features like punctuation, POS tags, and emoticons, and Misra and Arora [16] introduced an attention-based BiLSTM, which improved context awareness and keyword interpretation. Xiong et al. [17] proposed a self-matching mechanism within a BiLSTM to reduce redundancy while preserving accuracy. More recent studies rely heavily on transformer-based and hybrid models. Akula and Garibay [18] introduced a multi-head self-attention model, while Kumar et al. [19] combined RoBERTa embeddings with BiLSTMs. Other works experimented with rule-based feature ensembles [20], contextual BERT embeddings [21], or RCNN-RoBERTa hybrids [22]. Parameswaran et al. [23] investigated sarcasm targets using deep learning with aspect-based sentiment models, and Du et al. [24] combined semantics with emotional cues using a two-stream CNN. Hybrid BERT-LSTM models have also been applied to English and code-mixed tweets [25–27]. Researchers have also explored multimodal and user-specific signals. Hazarika et al. [28] incorporated user traits and behaviors, Illic et al. [29] used character-level embeddings with ELMo, and Malave and Dhage [30] analyzed user behavioral patterns. Hashtags [31] and multimodal approaches combining text with images [32] have also been explored. Emotion-based methods have been studied by Agrawal et al. [33], while cross-lingual sarcasm detection has been attempted for Arabic [35], Hindi-English [36], and other languages [34, 37–39]. Some advanced models specifically addressed new forms of sarcasm. Kamal and Abulaish [40] proposed a self-deprecating sarcasm detection model using CNNs with BiGRU and attention mechanisms, while Elkamchouchi et al. [41] introduced a hybrid autoencoder-based system optimized with cuckoo search. In summary, past research has shifted from simple ML classifiers based on lexical cues to deep learning and hybrid transformer-based approaches. However, many of these methods still rely heavily on either content or context alone, making them less generalizable across platforms. To handle these issues, the present work suggests a hybrid framework that integrates both word-level .and sentence-level embeddings, with a fuzzy logic layer to balance outputs and improve adaptability.

III. PROPOSED FRAMEWORK

This study introduces a hybrid ensemble framework designed to improve sarcasm detection in social media texts by combining both word level and sentence level contextual understanding. The framework integrates three key embedding models: Word2Vec**,** GloVe**,** and DistilBERT**.** each contributing a distinct perspective on the text's meaning[42]. While Word2Vec and GloVe focus on representing individual words based on their co-occurrence and proximity in large corpora, DistilBERT generates sentence level embeddings by processing entire input sequences with bidirectional attention. By bringing together these three representation techniques, the model can effectively analyze both fine-grained word relationships and broader contextual dependencies, which are crucial for detecting sarcasm. The architecture of the proposed model begins with a **preprocessing layer**, which handles tasks such as stop word removal, punctuation cleaning, and normalization[43]. Once the input is cleaned, the text is simultaneously passed into the Word2Vec, GloVe, and DistilBERT modules. Each of these modules transforms the input into a dense embedding vector: Word2Vec and GloVe produce word embeddings, while DistilBERT

generates a sentence-level embedding that captures contextual information across the entire input. These embeddings are then processed through separate dense layers, allowing each model to learn intermediate features tailored for the sarcasm detection task. The output from each dense layer is passed through a SoftMax classifier, which calculates the likelihood that the input text is sarcastic. The final classification decision is made by a fuzzy logic controller, which receives the probability scores from all three models. Unlike rigid voting mechanisms, this fuzzy layer interprets each model's output as a linguistic variable categorized as *high*, *medium*, or *low* and applies rule-based reasoning to synthesize a final prediction. For example, if all three models output a high sarcasm probability, the fuzzy logic system classifies the input as sarcastic with high confidence. If the outputs are mixed, such as one high and two medium scores, the decision will reflect a lower certainty. This rule-based fusion strategy allows the model to handle ambiguity more gracefully and improves interpretability compared to purely neural approaches.

The combination of content-based word embeddings and context-aware sentence embeddings, along with the fuzzy decision layer, gives this model a unique edge. It can detect sarcasm in text where meaning is often subtle, ironic, or dependent on contrast. GloVe and Word2Vec contribute strong lexical understanding, while DistilBERT offers deeper contextual interpretation. The fuzzy layer harmonizes their outputs, compensating for the weaknesses of each model when used individually. A detailed schematic of the model's data flow is shown in Figure 1, illustrating how the components interact from preprocessing to final classification.

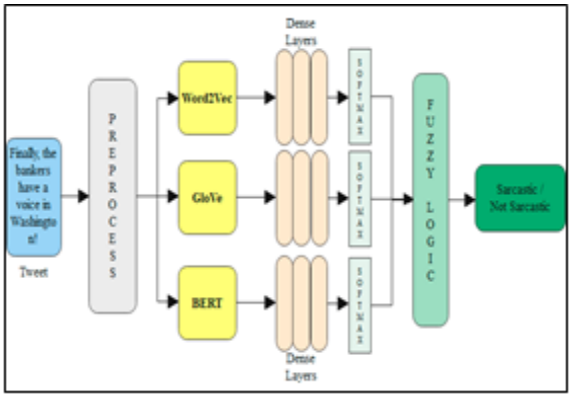

Figure 1. Architecture of the proposed model.

A. Word Vectorization

The process of turning textual data into numerical representations, or embeddings, that capture the semantic meaning and relationships between words is called word vectorization. This transformation's main objective is to reduce the text's high-dimensional representation into a more manageable and compact form while maintaining contextual similarities within the text. These embeddings serve as the basis for numerous natural language processing (NLP) methods, which allow computers to efficiently comprehend and evaluate human discourse. Word vectorization aids models in comprehending linguistic patterns and subtleties by teaching them how words relate to one another in a particular corpus. Word2Vec and GloVe are two well-liked methods in this field that are frequently employed to produce dense word embeddings that accurately represent the language's underlying semantics.

B. Global Vectors for Word Representation (GLOVE)

Global Vectors for Word Representation is a word embedding technique developed at Stanford that converts words into meaningful numerical vectors by learning from how often they occur together in a text corpus. It begins by building a co-occurrence matrix that records the frequency of word pairs, and then trains a regression model to produce low-dimensional, dense vectors that capture both the meaning of words and their relationships. The model works on two main ideas: nearest neighbors and linear substructures. Nearest neighbors are identified using cosine similarity or Euclidean distance, which measures how close two words are semantically, for example "doctor" being closer to "nurse" than "banana." Linear substructures go a step further by capturing analogies through vector differences, such as king – man + woman ≈ queen. Unlike simple similarity scores, GloVe also uses probability ratios to highlight distinctions; for example, the words *ice* and *steam* can be differentiated by how often they co-occur with probe words like *cold*, *hot*, *solid*, or *gas*. Thus, GloVe combines global co-occurrence statistics with local context to create word embeddings that preserve both semantic similarity and meaningful word relationships, making it a powerful tool for natural language processing.

Probability and Ratio	k = solid	k = water	k = gas	k = fashion
P(k|ice)	1.9×10^{-4}	6.6×10^{-5}	3.0×10^{-3}	1.7×10^{-5}
P(k|steam)	2.2×10^{-5}	7.8×10^{-4}	2.2×10^{-3}	1.8×10^{-5}
P(k|ice) / P(k|steam)	8.9	8.5×10^{-2}	1.36	0.96

Table 1. The table of probability ratios of GloVe working for a corpus of words.

GloVe is designed to capture meaningful relationships between words by using a word co-occurrence matrix, which records how often words appear together in a large corpus. It focuses on word-pair relationships rather than individual word interactions, helping it perform better than models like Word2Vec in tasks such as word analogies. Words are represented by the embeddings it generates in a fashion that reflects their semantic meaning and similarity. The co-occurrence matrix must be stored in a large amount of memory by GloVe, and it must frequently be rebuilt when model settings are changed. This can be computationally costly and time-consuming. The well-known word embedding method Word2Vec was created in 2013 by Tomas Mikolov and his colleagues at Google. From vast text datasets, it learns vector representations of words using shallow neural networks. Word2Vec is trained on a prediction task using either the Continuous Bag of Words (CBOW) model, which predicts a target word from its context, or the Skip-Gram model, which predicts surrounding words given a target word. This is in contrast to GloVe, which depends on global co-occurrence.

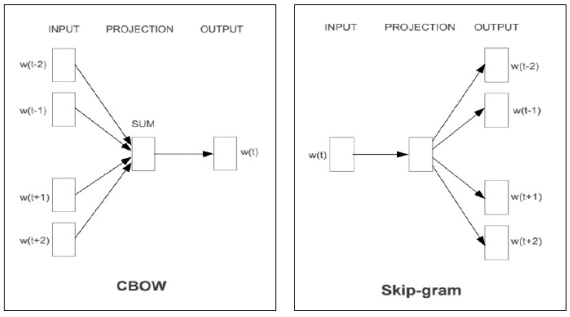

Figure 2. Two architectures, CBOW and Skip-Gram, are employed by Word2Vec

These embeddings capture both syntactic and semantic relationships between words, allowing them to be used in various NLP tasks like sentiment analysis, recommendation systems, and text classification. One of the key strengths of Word2Vec is its ability to group similar words closely in the vector space, enabling analogies like "king − man + woman ≈ queen." Word2Vec is memory efficient, can handle streaming data, and is easy to implement. While Skip-Gram works well with smaller datasets and rare words, CBOW is faster and performs better on frequent words. In our proposed framework, we use the CBOW model, as sarcastic expressions often include common terms that CBOW captures effectively.

C. Sentence Embedding

In order to capture a phrase's semantic content and the links between sentences for a variety of natural language processing (NLP) applications, including text categorization, similarity detection, and machine translation, the process of transforming a whole sentence into a numerical vector—referred to as a sentence vector—is known as sentence embedding. Simple techniques like concatenating or averaging word embeddings or sophisticated transformer-based models like BERT can be used to produce these embeddings. While earlier word embedding techniques like Word2Vec and GloVe produced useful word representations, they struggled with limitations such as handling out-of-vocabulary terms and distinguishing between opposite words like "good" and "bad," which often appeared too close in vector space and reduced their effectiveness in tasks like sentiment analysis. BERT (Bidirectional Encoder Representations from Transformers), developed by Google Research, overcomes these issues by using a self-supervised pretraining approach where parts of text are intentionally masked, and the model learns to predict them. Unlike traditional models that only looked at context in one direction using a sliding window, BERT leverages the transformer architecture to capture deep bidirectional context, processing both left and right words around a target simultaneously. This enhances BERT's capability to comprehend subtle meanings in sentences, and the fine-tuning of its pretrained model with a compact output layer facilitates exceptional performance across a diverse array of NLP applications. BERT-base is a robust language model architecture created by Google, featuring 12 encoder layers and generating 768-dimensional embeddings for input text. It is constructed on the transformer architecture and employs two primary training tasks: Masked

Language Modeling (MLM) and Next Sentence Prediction (NSP). MLM enables the model to acquire bidirectional context by randomly obscuring certain words in a sentence and training BERT to forecast them using context from both the left and right sides. NSP instructs the model to grasp sentence relationships by predicting whether one sentence logically follows another. During the training phase, half of the sentence pairs are sequential from the corpus, while the remaining half are random combinations.

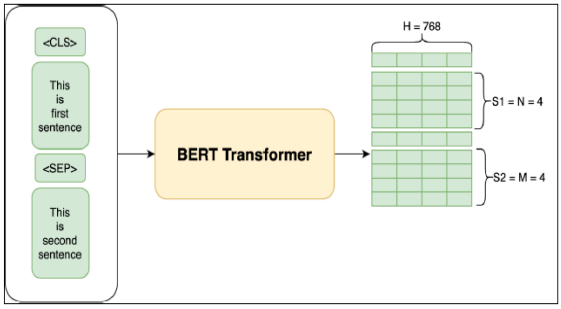

Figure 3. The architecture of BERT-base

D. Fuzzy Logic

Fuzzy logic is an extension of classical Boolean logic where the truth value of a statement can ranges continuously between 0.0 and 1.0, rather than being strictly true (1) or false (0). This approach allows reasoning over vague or imprecise information, making it especially useful in real-world applications where binary decisions are insufficient. For instance, a statement like “The person is tall” might hold a truth value of 0.9 instead of being absolutely true or false. A rule-based fuzzy logic system employs a collection of “if–then” rules that operate on input values mapped to fuzzy sets, defined by membership functions representing linguistic categories such as “low,” “medium,” and “high.” The fuzzy inference process combines the results of these rules and ultimately performs defuzzification to generate a crisp, final output. In the proposed sarcasm detection framework, fuzzy logic is utilized as the final decision-making module. The classification probabilities obtained from Word2Vec, GloVe, and DistilBERT in our adapted model are passed into the fuzzy logic controller. Each probability is categorized based on predefined thresholds: values above 0.75 are considered "high," between 0.40 and 0.75 as "medium," and below 0.40 as "low." These labels are then used in fuzzy rule evaluations to balance the strengths and weaknesses of the individual models, thereby enhancing overall prediction accuracy. This integration of fuzzy logic ensures interpretability, flexibility, and robustness in the final classification decision.

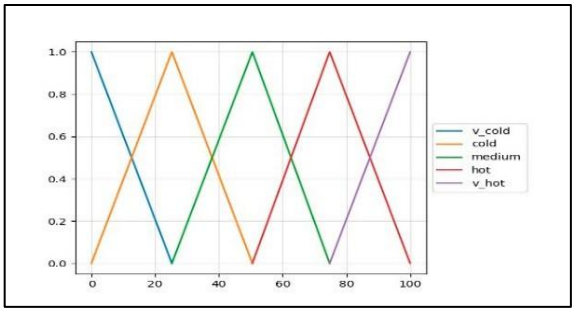

Figure 4. Membership function example for temperature

Fuzzy Logic Controller	Domain Knowledge	Examples of Fuzzy Rules
FLC_{Prob}	If a SoftMax probability value from Word2Vec is low, there is a low probability value from GloVe, and there is a low BERT value, then the final weighting unit from the fuzzy logic controller is likely to be low. Suppose a SoftMax probability value from Word2Vec is high. In that case, with a high probability value from GloVe, and a high BERT value, then the final weighting unit from the fuzzy logic controller is likely to be high.	If ($Word2Vec_{Prob}$ is low) and ($GloVe_{Prob}$ is low), and ($BERT_{Prob}$ is low), Then ($FuzzyLogicController_{Prob}$ is low). If ($Word2Vec_{Prob}$ is medium) and ($GloVe_{Prob}$ is medium) and ($BERT_{Prob}$ is medium), Then ($FuzzyLogicController_{Prob}$ is high). If ($Word2Vec_{Prob}$ is high) and ($GloVe_{Prob}$ is high) and ($BERT_{Prob}$ is high,), Then ($FuzzyLogicController_{Prob}$ is high).

Table 2. Fuzzy logic controller rules.

E. Execution of Framework

Following preprocessing, all comments, tweets, and headlines are processed through three components Word2Vec, GloVe, and BERT to create embeddings, with each model yielding distinct representations due to their reliance on different embedding techniques. These embeddings are subsequently input into dense layers for feature learning, and the outputs are directed to a SoftMax layer that produces probabilities indicating whether the text is authentic or fabricated. The resulting classification probabilities (Word2VecProb, GloVeProb, and BERTProb) are then integrated within a fuzzy logic module, which employs a rule-based methodology to evaluate them and arrive at a final determination regarding the text's sarcasm. To assess the model's performance, experiments were carried out on publicly accessible datasets such as SARC, Twitter datasets, and Headlines datasets. The model was developed utilizing Google’s TensorFlow framework with Keras, applying ReLU as the activation function and Adam as the optimizer with a learning rate of 3e-4. Given that the task involves binary classification,

binary cross-entropy was utilized as the loss function, with an 80-20 division for training and testing. Hyperparameters were maintained consistently across all dense layers, and training was concluded once the maximum accuracy was attained, with the optimal results documented using accuracy as the principal evaluation metric.

IV. DATASET

To measure how effectiveness of the suggested sarcasm detection framework, three publicly accessible datasets were employed: a Twitter dataset, the SARC 2.0 dataset sourced from Reddit, and a Headlines dataset gathered from online news outlets. These datasets encompass a variety of user-generated content domains, thereby providing a wide array of sarcasm styles and contexts.

A. Twitter Dataset

This dataset was built using publicly available tweets that were manually labeled for sarcasm. It includes both standalone tweets and user responses, allowing the model to learn from different conversational context. The dataset consists of 2,100 tweets**,** out of which 1,200 were marked as non-sarcastic and 900 as sarcastic**,** based on user annotations and contextual clues.

B. Sarc Dataset

The Self-Annotated Reddit Corpus (SARC 2.0) consists of sarcastic remarks collected from Reddit discussions. On Reddit, users frequently employ the /s token to signify sarcasm, which served as the foundation for labeling. This research utilized only the original top-level comments—excluding responses or parent-child relationships—to ensure consistency. Both the general and political segments of the dataset were incorporated. The dataset comprised around 20,000 comments, featuring an even distribution of sarcastic and non-sarcastic entries.

C. Headlines Dataset

This dataset combines headlines from two different news sources: The Onion**,** which publishes satirical articles, and HuffPost**,** which presents factual news content. Headlines from The Onion were treated as sarcastic, while those from HuffPost were considered genuine. The dataset includes a total of **30,000** headlines, of which 13,500 were identified as sarcastic **and** 16,500 as non-sarcastic**.**

D. Evaluation Metrics

To evaluate the performance of the proposed sarcasm detection model, we employed four standard classification metrics: accuracy, precision, recall, and F1-score. These metrics are widely accepted for evaluating binary classifiers and help measure both the correctness and completeness of the predictions. Accuracy provides the overall correctness of the model, while precision measures the proportion of correctly predicted positive cases among all predicted positives. Recall calculates the proportion of actual positives that were correctly identified, and the F1-score offers a harmonic mean of precision and recall, especially useful in imbalanced datasets. The formulas used for these metrics are as follows:

- Precision = TP / (TP + FP)
- Recall = TP / (TP + FN)
- Accuracy = (TP + TN) / (TP + TN + FP + FN)
- F1-score = 2 × (Precision × Recall) / (Precision + Recall)

Here, TP, TN, FP, and FN represent True Positives, True Negatives, False Positives, and False Negatives, respectively. Our proposed model was tested on three datasets Headlines**,** SARC, and Twitter achieving accuracy scores of 91.10%**,** 83.25%**,** and 87.40%**,** respectively. These results demonstrate the model's superior ability to detect sarcasm compared to earlier methods.

E. Result Analysis

The experimental assessment was conducted utilizing three benchmark datasets: SARC, Twitter, and Headlines, each comprising diverse forms of sarcastic content sourced from various social media platforms. The suggested hybrid model, which combines DistilBERT for contextual representation with Word2Vec, GloVe, and a fuzzy logic-based decision layer, surpassed numerous state-of-the-art models across all datasets.On the SARC dataset, the proposed model attained an accuracy of 83.25%, a precision of 0.8725, a recall of 0.8691, and an F1-score of 0.8708. This demonstrates a notable enhancement over earlier models such as RCNN-RoBERTa (79.00% accuracy) and CASCADE (74.00% accuracy). Regarding the Twitter dataset, the hybrid model reached an accuracy of 87.40%, a precision of 0.8824, a recall of 0.9041, and an F1-score of 0.8931. These findings indicate superior performance relative to previous methods like the

Sarcasm Magnet model (72.5% accuracy) and Multi-Head Attention (81.2% accuracy). For the Headlines dataset, the model achieved its peak accuracy of 91.10%, along with a precision of 0.9431, a recall of 0.9314, and an F1-score of 0.9372. This affirms the model's strength in detecting sarcasm in brief, headline-style text formats. The integration of semantic embeddings Word2Vec, GloVe with profound contextual comprehension DistilBERT and fuzzy rule-based classification facilitated enhanced generalization, diminished ambiguity in sarcastic expressions, and improved precision-recall trade-offs. These outcomes substantiate the efficacy of the proposed methodology in capturing both superficial and profound linguistic indicators of sarcasm.

V CONCLUSION

In this study, a hybrid model for sarcasm detection was introduced, which integrates distilbert, word2vec, glove, and a classification layer based on fuzzy logic to advance the detection of sarcasm across various social media platforms. The combination of deep contextual embeddings with traditional semantic vectors enabled the model to effectively recognize both implicit and explicit indicators of sarcasm. Experimental findings on three publicly accessible datasets sarc, twitter, and headlines showed that the proposed method consistently surpassed existing state-of-the-art models regarding accuracy, precision, recall, and f1-score. The model obtained an accuracy of 83.25% on the SARC dataset, 87.40% on Twitter, and 91.10% on the Headlines dataset, thereby confirming its robustness and adaptability to different types of quicker computations while maintaining a high level of contextual understanding, and the fuzzy logic layer improved the interpretability of decisions. Overall, the findings affirm the effectiveness of the ensemble approach in sarcasm detection and indicate its potential for implementation in real-time social media monitoring systems or sentiment-aware chatbots.

REFERENCES

[1] Edwards, V.V. Sarcasm: What It Is and Why It Hurts Us. 2014. Available online: https://www.scienceofpeople.com/sarcasm-why-it-hurts-us/ (accessed on 5 October 2021).

[2] Rothermich, K.; Ogunlana, A.; Jaworska, N. Change in humor and sarcasm use based on anxiety and depression symptom severity during the COVID-19 pandemic. *J. Psychiatr. Res.* 2021, *140*, 95–100.

[3] Ezaiza, H.; Humayoun, S.R.; Al Tarawneh, R.; Ebert, A. Person-vis: Visualizing personal social networks (ego networks). In Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems, San Jose, CA, USA, 7–12 May 2016; pp. 1222–1228.

[4] Akula, R.; Garibay, I. Viztract: Visualization of complex social networks for easy user perception. *Big Data Cogn. Comput.* 2019, *3*, 17.

[5] Singh, B.; Sharma, D.K. Predicting image credibility in fake news over social media using multi-modal approach. *Neural Comput. Appl.* 2021, *34*, 21503–21517.

[6] Singh, B.; Sharma, D.K. SiteForge: Detecting and localizing forged images on microblogging platforms using deep convolutional neural network. *Comput. Ind. Eng.* 2021, *162*, 107733.

[7] Wallace, B.C. Computational irony: A survey and new perspectives. *Artif. Intell. Rev.* 2015, *43*, 467–483.

[8] Khodak, M.; Saunshi, N.; Vodrahalli, K. A large self-annotated corpus for sarcasm. In Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC), Miyazaki, Japan, 7–12 May 2018.

[9] Eke, C.I.; Norman, A.A.; Shuib, L.; Nweke, H.F. Sarcasm identification in textual data: Systematic review, research challenges and open directions. *Artif. Intell. Rev.* 2020, *53*, 4215–4258.

[10] Sarsam, S.M.; Al-Samarraie, H.; Alzahrani, A.I.; Wright, B. Sarcasm detection using machine learning algorithms in Twitter: A systematic review. *Int. J. Mark. Res.* 2020, *62*, 578–598.

[11] Keerthi Kumar, H.M.; Harish, B.S. Sarcasm classification: A novel approach by using Content Based Feature Selection Method.

[12] *Procedia Comput. Sci.* 2018, *143*, 378–386.

[13] Pawar, N.; Bhingarkar, S. Machine Learning

based Sarcasm Detection on Twitter Data. In Proceedings of the 5th International Conference on Communication and Electronics Systems (ICCES), Coimbatore, India, 10–12 June 2020.

[14] Ghosh, A.; Veale, T. Magnets for sarcasm: Making sarcasm detection timely, contextual and very personal. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, Copenhagen, Denmark, 7–11 September 2017;

[15] pp. 482–491.

[16] Ghosh, D.; Fabbri, A.S.; Muresan, S. Sarcasm analysis using conversation context. *Comput. Linguist.* 2018, *44*, 755–792.

[17] Liu, L.; Priestley, J.L.; Zhou, Y.; Ray, H.E.; Han, M. A2text-net: A novel deep neural network for sarcasm detection. In Proceedings of the IEEE First International Conference on Cognitive Machine Intelligence (CogMI), Los Angeles, CA, USA, 12–14 December 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 118–126.

[18] Misra, R.; Arora, P. Sarcasm detection using hybrid neural network. *arXiv* 2019, arXiv:1908.07414.

[19] Xiong, T.; Zhang, P.; Zhu, H.; Yang, Y. Sarcasm detection with self-matching networks and low-rank bilinear pooling. In Proceedings of the World Wide Web Conference, San Francisco, CA, USA, 13–17 May 2019; pp. 2115–2124.

[20] Akula, R.; Garibay, I. Interpretable Multi-Head Self-Attention Model for Sarcasm. Detection in Social Media. *Entropy* 2021, *23*, 394.

[21] Kumar, A.; Narapareddy, V.T.; Srikanth, V.A.; Malapati, A.; Neti, L.B.M. Sarcasm Detection Using Multi-Head Attention Based Bidirectional LSTM. *IEEE Access* 2020, *8*, 6388–6397.

[22] Sundararajan, K.; Palanisamy, A. Multi-Rule Based Ensemble Feature Selection Model for Sarcasm Type Detection in Twitter.

[23] *Comput. Intell. Neurosci.* 2020, *2020*, 2860479.

[24] Babanejad, N.; Davoudi, H.; An, A.; Papagelis, M. Affective and Contextual Embedding for Sarcasm Detection. In Proceedings of the 28th International Conference on Computational Linguistics, Barcelona, Spain, 8–13 December 2020.

[25] Potamias, R.A.; Siolas, G.; Stafylopatis, A. A transformer-based approach to irony and sarcasm detection. *Neural Comput. Appl.*

[26] 2020, *32*, 17309–17320.

[27] Parameswaran, P.; Trotman, A.; Liesaputra, V.; Eyers, D. Detecting the target of sarcasm is hard: Really? *Inf. Process. Manag.* 2021,

[28] *58*, 102599.

[29] Du, Y.; Li, T.; Pathan, M.S.; Teklehaimanot, H.K.; Yang, Z. An Effective Sarcasm Detection Approach Based on Sentimental Context and Individual Expression Habits. *Cogn. Comput.* 2021, *14*, 78–90.

[30] Sharma, D.K.; Singh, B.; Agarwal, S.; Kim, H.; Sharma, R. Sarcasm Detection over Social Media Platforms Using Hybrid Auto-Encoder-Based Model. *Electronics* 2022, *11*, 2844.

[31] Pandey, R.; Singh, J.P. BERT-LSTM model for sarcasm detection in code-mixed social media post. *J. Intell. Inf. Syst.* 2022.

[32] Savini, E.; Caragea, C. Intermediate-Task Transfer Learning with BERT for Sarcasm Detection. *Mathematics* 2022, *10*, 844.

[33] Hazarika, D.; Poria, S.; Gorantla, S.; Cambria, E.; Zimmermann, R.; Mihalcea, R. Cascade: Contextual sarcasm detection in online discussion forums. In Proceedings of the 27th International Conference on Computational Linguistics, Santa Fe, NM, USA, 20–26 August 2018; pp. 1837–1848.

[34] Ilic, S.; Marrese-Taylor, E.; Balazs, J.A.; Matsuo, Y. Deep contextualized word representations for detecting sarcasm and irony. In Proceedings of the 9th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis, Brussels, Belgium, 31 October 2018; pp. 2–7.

[35] Malave, N.; Dhage, S.N. Sarcasm Detection on Twitter: User Behavior Approach. In *Intelligent Systems, Technologies and Applications*;

[36] Advances in Intelligent Systems and Computing; Springer: Singapore, 2020; Volume 910.

[37] Sykora, M.; Elayan, S.; Jackson, T.W. A qualitative analysis of sarcasm, irony and related #hashtags on Twitter. *Big Data Soc.* 2020,

[38] *7*, 2053951720972735.

[39] Yao, F.; Sun, X.; Yu, H.; Zhang, W.; Liang, W.; Fu, K. Mimicking the Brain's Cognition of Sarcasm From Multidisciplines for Twitter Sarcasm Detection. *IEEE Trans. Neural Netw. Learn. Syst.* 2021, *34*, 228–242.

[40] Agrawal, A.; An, A.; Papagelis, M. Leveraging Transitions of Emotions for Sarcasm Detection. In Proceedings of the 43rd Interna- tional ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual, 25–30 July 2020; pp. 1505–1508.

[41] Techentin, C.; Cann, D.R.; Lupton, M.; Phung, D. Sarcasm detection in native English and English as a second language speakers.

[42] B.Dilip Kumar Reddy,A Student Evaluation Tool By Natural Language Processing,Anveshana's International Journal Of Research in Engineering and Applied Sciences

[43] B.Dilip Kumar Reddy,Deep Stock Prediction using Visual Interpretation: DeepClue, International Journal of Recent Technology and Engineering.

[44] *Can. J. Exp. Psychol./Rev. Can. Psychol. Expérimentale* 2021, *75*, 133–138.

[45] Farha, I.A.; Magdy, W. From Arabic Sentiment Analysis to Sarcasm Detection: The ArSarcasm Dataset. In Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection, Marseille, France, 12 May 2020.

[46] Swami, S.; Khandelwal, A.; Singh, V.; Akhtar, S.S.; Shrivastava, M. A Corpus of English-Hindi Code-Mixed Tweets for Sarcasm Detection. *arXiv* 2018, arXiv:1805.11869v1.

[47] Pradhan, R.; Agarwal, G.; Singh, D. Comparative Analysis for Sentiment in Tweets Using LSTM and RNN. In *International Conference on Innovative Computing and Communications*; Khanna, A., Gupta, D., Bhattacharyya, S., Hassanien, A.E., Anand, S., Jaiswal, A., Eds.; Advances in Intelligent Systems and Computing; Springer: Singapore, 2022; Volume 1387.

[48] Pradhan, R. Extracting Sentiments from YouTube Comments. In Proceedings of the 2021 Sixth International Conference on Image Information Processing (ICIIP), Shimla, India, 26–28 November 2021; pp. 1–4.

[49] Jain, V.; Agrawal, M.; Kumar, A. Performance Analysis of Machine Learning Algorithms in Credit Cards Fraud Detection. In Proceedings of the 2020 8th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO), Noida, India, 4–5 June 2020; pp. 86–88.

[50] Kamal, A.; Abulaish, M. CAT-BiGRU: Convolution and Attention with Bi-Directional Gated Recurrent Unit for Self-Deprecating Sarcasm Detection. *Cogn. Comput.* 2022, *14*, 91–109.

[51] Elkamchouchi, D.H.; Alzahrani, J.S.; Asiri, M.M.; Al Duhayyim, M.; Mohsen, H.; Motwakel, A.; Zamani, A.S.; Yaseen, I. Hosted Cuckoo Optimization Algorithm with Stacked Autoencoder-Enabled Sarcasm Detection in Online Social Networks. *Appl. Sci.* 2022, *12*, 7119.

[52] Pennington, J.; Socher, R.; Manning, C. GloVe: Global Vectors for Word Representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 25–29 October 2014; Association for Computational Linguistics: Toronto, ON, Canada, 2014; pp. 1532–1543.

[53] Goyal, C. Part 6: Step by Step Guide to Master NLP—Word2Vec. Available online: https://www.analyticsvidhya.com/blog/2021

[54] /06/part-6-step-by-step-guide-to-master-nlp-word2vec/ (accessed on 1 October 2021).

[55] Vu, K. BERT Transformers: How Do They Work? 2021. Available online: https://dzone.com/articles/bert-transformers-how-do- they-work (accessed on 1 January 2023).

[56] Singhala, P.; Shah, D.; Patel, B. Temperature Control using Fuzzy logic. *arXiv* 2014, arXiv:1402.3654.

[57] Sahu, G.A.; Hudnurkar, M. Sarcasm Detection: A Review, Synthesis and Future Research Agenda. *Int. J. Image Graph.* 2022.