IJNRD.ORG ISSN : 2456-4184
INTERNATIONAL JOURNAL OF NOVEL RESEARCH AND DEVELOPMENT (IJNRD) | IJNRD.ORG
An International Open Access, Peer-reviewed, Refereed Journal

SARCASM DETECTION IN TEXT

[1]Nagendra NP , [2]Ashritha R Murthy

[1]PG Student, [2]Assistant Professor
[1]Computer Science and Engineering,
[1]JSS Science and Technology University, Mysuru, India

Abstract : Sarcasm is a kind of word or speech where the intended meaning differs from the literal expression. Detecting sarcasm in a text is a significant challenge in Natural Language Processing as it requires contextual, semantic, and sentiment awareness. Failure to identify sarcasm leads to misleading outcomes in sentiment analysis, opinion mining, and customer feedback analysis. This research proposes a sarcasm detection system that integrates Machine Learning (Logistic Regression, Naïve Bayes, SVM), Deep Learning (Convolutional Neural Networks), and transformer-based architectures (BERT). Three datasets were employed: Sarcasm Headlines Dataset (Dataset 1), Sarcasm Headlines Dataset v2 (Dataset 2), and a Twitter-based sarcasm dataset (Dataset 3) Preprocessing steps included text normalization, cleaning, tokenization, part-of-speech tagging, and handling class imbalance using Random Oversampling Exploratory data analysis involved word clouds, POS tag distribution, and frequency analysis to highlight patterns of sarcastic vs. non-sarcastic text. Experimental results show that BERT got the high performance with F1-score above 0.90, outperforming CNN and traditional ML models. CNN, however, demonstrated competitive results with lower computational requirements, while Logistic Regression, Naïve Bayes, and SVM served as effective baselines. The proposed framework validates the potential of combining classical and modern NLP methods to improve sarcasm detection accuracy.

IndexTerms **- Sarcasm detection, Natural Language Processing, Machine Learning, CNN, BERT, Sentiment Analysis.**

1. INTRODUCTION

Sarcasm represents a form of speech or verbal irony in which the intended meaning differs from the literal wording [1]. This linguistic phenomenon poses an important challenge in NLP, particularly within sentiment analysis, because sarcastic remarks often reverse apparent polarity and mislead automated systems [19]. In today's modern world, it is essential to identify human emotions in text data, since many people rely on text-based communication instead of face-to-face interaction. Sarcasm adds a further layer of complexity, making it challenging even for humans and more complicated for machines [2].

Unlike literal text, sarcasm is highly context-dependent, often requiring an understanding of subtle cues such as contrast between expectation and reality, tone implied by punctuation, and exaggeration [8]. Traditional machine learning techniques using bag-of-words or TF-IDF often fail to capture these deeper relationships, while deep learning and transformer-based models such as BERT provide stronger contextual representations [15].

Moreover, sarcasm detection is not only relevant for sentiment analysis but also has applications in opinion mining, online safety, customer feedback monitoring, and even in detecting misinformation or harmful discourse. A successful system must be able to distinguish between genuine positive statements and sarcastic remarks that carry negative undertones. This involves integrating semantic, syntactic, and pragmatic features of language to improve model accuracy.

Recent research has also shown the importance of multimodal signals—such as emojis, hashtags, and even images—when sarcasm occurs in social media conversations. These additional layers of meaning can provide strong cues that are otherwise lost in plain text. Therefore, combining textual data with auxiliary signals has the potential to enhance detection accuracy significantly. This research integrates classical ML, deep learning, and transformer-based methods to build a robust sarcasm detection pipeline [1]. By leveraging both statistical and contextual approaches, the study aims to address limitations of earlier techniques and contribute to more reliable emotion-aware natural language processing systems.

2. LITERATURE REVIEW

Over the past decennary, sarcasm detection research have shifted from handcrafted feature-based systems to deep learning and transformer-driven approaches.

Early methods relied on specific indicate such as n-grams, sentiment lexicons, and manually designed contextual rules. Sinha and Choudhary [1] surveyed these approaches, noting their limited scalability and inability to capture implicit sarcasm. Razali et al. [2] highlighted the importance of incorporating contextual information, showing that models using context-aware embeddings consistently outperform bag-of-words baselines.

Yaghoobian et al. [3] compared a wide range of algorithms and confirmed that deep learning models significantly outperform conventional ML classifiers in capturing sarcasm's complexity. Aa et al. [4] broadened this view by discussing challenges such as cultural variation and implicit sarcasm. In a focused study, Šandor and Bagić Babac [5] examined sarcasm detection in online comments, finding that ML-based systems achieved reasonable accuracy but faced limitations in noisy environments like social media.

With the increase of deep learning, Chowdary et al. [6] demonstrated that CNN models outperform statistical approaches on user-generated text. Kumar and Gupta [7] explored hybrid CNN-RNN architectures for Reddit sarcasm detection, while Hazarika et al. [8] developed CASCADE, a conversationally aware model designed for discussion forums.

Beyond single-modality systems, multimodal approaches gained momentum. Khandelwal and Bedi [9] integrated audio and text features, Oprea and Magdy [10] used user embeddings and contextual cues, and Kumar and Ramakrishnan [11] employed ensemble learning with emotion features.Vempala and Preotiuc-Pietro [12] examined the role of visual elements in shaping how sarcasm is understood within tweets, whereas Castro and Lupu [13] introduced MUStARD, a multimodal dataset created to support sarcasm detection in conversational settings.

The availability of large datasets also facilitated contextual learning. Bamman et al. [14] analyzed Reddit sarcasm with contextual embeddings, while Agrawal and Annamalai [15] demonstrated that transformers with tweet embeddings achieve state-of-the-art results. Naskar and Dey [16] introduced capsule networks to model hierarchical sarcasm cues, and Mahajan and Patel [17] compared different transformer variants, finding BERT-like architectures to be superior.

Alternative approaches have also emerged. Chopra and Bansal [18] applied fuzzy logic to detect sarcasm and irony in Hindi-English code-mixed text, while Nasir et al. [19] surveyed ML and DL approaches, highlighting ongoing issues such as domain adaptation and implicit sarcasm. Zhang and Liu [20] improved BERT's robustness by applying adversarial training techniques.

Overall, three trends are evident: (i) the progression from shallow ML to deep and transformer-based methods, (ii) the inclusion of contextual and multimodal information, and (iii) the exploration of robust and hybrid frameworks for improved generalization.

3. METHODOLOGY

The methodology adopted in this study integrates **data collection, preprocessing, Performance Measures, model implementation, training, and evaluation**. By leveraging a combination of **traditional (ML), learning (DL), and transformer-based models**, the system aims to capture both lexical and contextual nuances of sarcasm in text.

3.1. Research Framework

The overall research framework is illustrated in Fig. 1 (Methodology Flowchart). The system is divided into five sequential stages:

1. **Dataset Acquisition** – Collecting benchmark sarcasm datasets from reliable sources such as Kaggle and Twitter repositories.
2. **Data Preprocessing** – Cleaning and transforming raw text to ensure high-quality input for modeling.
3. **Feature Extraction** – Generating numerical representations of text through statistical and contextual embedding techniques.
4. **Model Training** – Applying multiple ML, CNN, and transformer-based models to learn sarcasm patterns.
5. **Evaluation and Comparative Analysis** – Measuring performance using statistical metrics and visualizations.

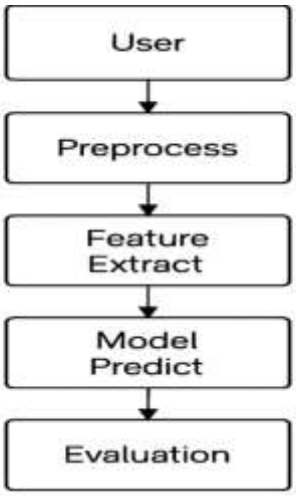

Figure 1: Data Flow Diagram

3.2. Datasets

To ensure diversity and robustness, three publicly available sarcasm datasets were employed:

- **Dataset 1 – Sarcasm Headlines Dataset (Kaggle):** Contains approximately 26,709 labeled news headlines. Each headline is annotated as sarcastic or non-sarcastic, making it suitable for formal text analysis.
- **Dataset 2 – Sarcasm Headlines Dataset v2:** An extended version of Dataset 1, covering additional headlines and improved balance between classes.
- **Dataset 3 – Twitter Sarcasm Dataset:** Comprises roughly 28,548 tweets annotated for sarcasm, providing a rich resource for informal, conversational sarcasm.

Total Size After Combining: 92,528 samples. Together, these datasets capture both **formal text (news domain)** and **informal text (social media)**. Class imbalance, a common limitation in sarcasm detection [5], was addressed through **Random Over Sampling (ROS)** to equalize sarcastic and non-sarcastic samples, ensuring fairness in training.

3.3. Data Preprocessing

Preprocessing is crucial in sarcasm detection because sarcastic cues are subtle and often hidden in punctuation, exaggeration, or wordplay. The following preprocessing steps were employed:

1. **Noise Removal** – Eliminated irrelevant elements such as hyperlinks, emojis, hashtags, and HTML tags that do not contribute to semantic meaning.
2. **Case Normalization** – Converted all text to lowercase to maintain consistency.
3. **Retention of Sarcasm Markers** – Unlike traditional preprocessing where punctuation is discarded, this study retained markers like *"!", "...", and "?"* because they often signal sarcasm [1].
4. **Tokenization** – Split sentences into tokens using the NLTK tokenizer for ML/CNN models and the BERT tokenizer for DistilBERT.
5. **Padding and Truncation** – Ensured uniform input length. CNN inputs were padded/truncated to a maximum length of 100 tokens, while DistilBERT inputs were limited to 128 tokens.
6. **Balancing** – Applied Random Over Sampling (ROS) to address skewed class distributions. This was essential to prevent bias toward the majority (non-sarcastic) class.

3.4. Feature Extraction

In this study, different feature extraction techniques were employed according to the model type. Classical machine learning models, including Logistic Regression, Naïve Bayes, and Support Vector Machine (SVM), utilized Term Frequency–Inverse Document Frequency vectors, which assign weights to words based on their frequency across documents, thereby emphasizing terms that are particularly discriminative for sarcasm detection. For the Convolutional Neural Network (CNN), word embeddings were employed to represent words as dense vectors that capture semantic relationships. Pre-trained embeddings, such as GloVe or Word2Vec, were utilized to provide rich and informative representations of words. In the transformer-based model, DistilBERT was used to generate contextual embeddings, offering dynamic, context-aware representations. Unlike TF-IDF or static embeddings, BERT captures nuanced meanings based on sentence structure, significantly improving performance in detecting sarcasm where context is critical.

3.5. Model Implementation

3.5.1. Machine Learning Models:

Logistic Regression was employed as the benchmark model, utilizing TF-IDF representations to capture words and phrases associated with sarcasm. Multinomial Naïve Bayes, a probabilistic model stranded in Bayes' theorem, was applied for it's suitability with sparse text data, while SVM with a linear kernel effectively managed the high-dimensional TF-IDF features by identifying optimal decision boundaries. These models provided a benchmark for evaluating more complex deep learning and transformer-based approaches.

3.5.2. Deep Learning Model (CNN):

The CNN architecture was designed to capture local patterns and phrases indicative of sarcasm. The network took word embeddings as input, followed by a one-dimensional convolutional layer to detect n-gram features. A global max pooling layer reduced dimensionality, while a dense layer represented semantic information. Dropout layers were incorporated to prevent overfitting, and a sigmoid-activated output layer was used for binary classification. CNNs are particularly effective at detecting contrastive word pairs or exaggerated sentiment patterns, which are common in sarcastic expressions.

3.5.3. Transformer Model (BERT):

BERT was trained on sarcasm datasets using the HuggingFace Transformers library. Input to the model included token IDs and attention masks, and a classification layer was added on top of the transformer encoder. DistilBERT offers the advantage of capturing long-range dependencies and bidirectional context, enabling it to detect subtle sarcasm patterns that conventional CNN or LSTM models might miss. This contextual understanding is particularly important when sarcasm relies on sentence structure or contrastive cues.

3.6. Model Training

This was done using an 80–20 train-testing split, with the training set used to optimize model parameters and the test-set for evaluating generalization. CNN was trained with a batch size of 32, while BERT employed a batch size of 16 due to GPU memory constraints. Both models were trained for four epochs, balancing accuracy with overfitting prevention. The binary class-entropy loss function was used for CNN and classical machine learning models, while sparse categorical cross-entropy was applied for BERT. The Adam optimizer with adaptive learning rate scheduling was employed to ensure stable and efficient convergence, optimizing performance without incurring excessive computational cost.

3.7. Evaluation Metrics

Model performance was calculated using a comprehensive set of results to ensure robust assessment beyond simple accuracy. **Accuracy** measures the overall accurate of predictions and is calculated as:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \quad \text{---------- 1}$$

where, TP represents true positives, TN-true negatives, FP-false positives, and FN -false negatives[1]. **Precision** indicates the proportion of correctly predicted sarcastic sentences among all sentences predicted as sarcastic

$$\text{Precision} = \frac{TP}{TP + FP} \quad \text{---------- 2}$$

Recall (or sensitivity) measures the ability of the model to properly identify all original sarcastic sentences:

$$\text{Recall} = \frac{TP}{TP + FN} \quad \text{---------- 3}$$

F1-score is the harmonic mean of precision and recall, providing a single metric that balances both measures, particularly useful in imbalanced datasets :

$$\text{F1-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad \text{---------- 4}$$

In addition, **confusion matrices** were analyzed to gain detailed insights into misclassifications, highlighting both false positives, where non-sarcastic sentences were incorrectly labeled as sarcastic, and false negatives, where sarcasm was missed. Together, these metrics offered a holistic framework for evaluating model performance in sarcasm detection.

3.8. Visualization and Analysis

Exploratory analysis was carried out to support the quantitative findings. Separate word clouds were created for sarcastic and non-sarcastic instances, which illustrated clear contrasts in the vocabulary used (Sinha & Choudhary, 2023) [1]. Part-of-speech (POS)

distribution analysis further showed that sarcastic texts tend to include a higher proportion of adjectives and adverbs than literal statements (Šandor & Bagić Babac, 2021) [5]. Moreover, frequency-based plots revealed the recurrence of typical sarcasm cues such as "yeah right," "totally," and "obviously," aligning with earlier studies that emphasized lexical markers as reliable indicators (Chowdary et al., 2023) [6]. These visualization methods offered interpretable insights into the linguistic patterns of sarcasm, thereby validating the model outputs and reinforcing the significance of observable lexical cues in sarcasm detection (Yaghoobian et al., 2021) [3].

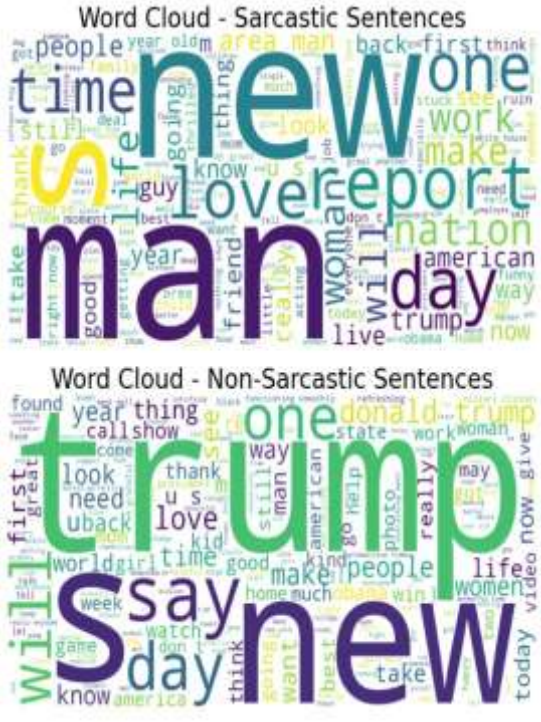

Figure 2: Word cloud for sarcastic and non-sarcastic.

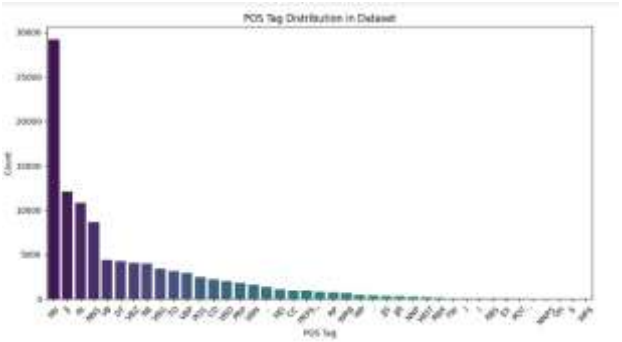

Figure 3: Part-of-speech (POS) tagging.

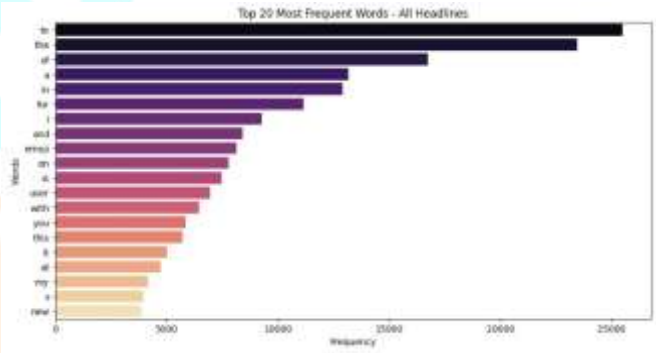

Figure 4: word frequency graph.

4. RESULTS AND DISCUSSION

The performance of five models—**Logistic Regression, Naïve Bayes, SVM, CNN, and BERT**—was evaluated on three individual datasets and the combined dataset. The evaluation metrics considered were **Accuracy, Precision, Recall, and F1-Score**.

Table 4.1: Model Evaluation on dataset1

Model	Accuracy	Precision	Recall	F1-Score
Logistic Regression	0.869931	0.831325	0.882805	0.856292
Naive Bayes	0.838968	0.802905	0.839133	0.820620
SVM	0.875697	0.840408	0.884852	0.862058
CNN	0.910704	0.872161	0.933385	0.901735
BERT	**0.965293**	**0.957849**	**0.963323**	**0.960578**

From Table 4.1, it is observed that all models perform relatively well, with accuracy values ranging between **83% and 96%**. Among classical machine learning models, **SVM and Logistic Regression** demonstrated competitive results, achieving accuracies of **87.36%** and **86.99%**, respectively. **CNN outperformed the classical models**, with an accuracy of **91.01%** and a high recall of **93.38%**, indicating its strength in correctly identifying sarcastic samples. The **BERT model achieved the best overall performance**, with an accuracy of **96.52%**, precision of **95.78%**, recall of **96.33%**, and an F1-score of **96.06%**, demonstrating the advantage of transformer-based contextual understanding.

Table 4.2: Model Evaluation on dataset2

Model	Accuracy	Precision	Recall	F1-Score
Logistic Regression	0.868409	0.850725	0.877805	0.864053
Naive Bayes	0.836158	0.824682	0.833211	0.828925
SVM	0.873965	0.858953	0.879933	0.869316
CNN	0.909116	0.887367	0.926874	0.906691
BERT	**0.962298**	**0.963318**	**0.957313**	**0.960306**

In Table 4.2, a similar performance trend was observed. Classical models achieved accuracies between **83.65% and 87.39%**, with SVM again outperforming Naïve Bayes and Logistic Regression in terms of precision (**85.85%**). **CNN maintained high accuracy (90.91%) and recall (92.68%)**, further validating its effectiveness for this task.**BERT once again outperformed all other models**, achieving an accuracy of **96.22%** and an F1-score of **96.03%**, showing its robustness across multiple datasets.

Table 4.3: Model Evaluation on dataset3

Model	Accuracy	Precision	Recall	F1-Score
Logistic Regression	0.771433	0.744051	0.716344	0.729935
Naive Bayes	0.755829	0.714423	0.722578	0.718477
SVM	0.780962	0.762889	0.713916	0.737590
CNN	0.804105	0.747886	0.823201	0.783738
BERT	**0.860688**	**0.854984**	**0.815187**	**0.834611**

Evaluation on Dataset 3 (Table 4.3) shows lower overall performance compared to the previous datasets. Logistic Regression and Naïve Bayes achieved accuracies of **77.14%** and **75.59%**, respectively, indicating difficulties in handling dataset variations. SVM performed slightly better (**78.09% accuracy**) but still struggled with recall (**71.35%**).**CNN performed better than classical models**, with an accuracy of **80.41%** and F1-score of **78.37%**, demonstrating its adaptability. However, **BERT remained the best-performing model**, with an accuracy of **86.06%** and F1-score of **83.46%**, though its performance was lower than on the previous datasets. This suggests Dataset 3 may have more challenging or noisy samples.

Table 4.4: Model Evaluation of Combined dataset

Model	Accuracy	Precision	Recall	F1-Score
Logistic Regression	0.817951	0.824544	0.807143	0.815751
Naive Bayes	0.792446	0.797794	0.782684	0.790167
SVM	0.823138	0.832053	0.809091	0.820412
CNN	0.829677	0.817149	0.848810	0.832679
BERT	0.904950	0.915196	0.892316	0.903611

When all three datasets were merged (Table 4.4), the models' performances showed a balanced trend. Logistic Regression, Naïve Bayes, and SVM achieved accuracies of **81–82%**, with F1-scores around **0.79–0.82**, confirming their consistency across datasets. **CNN again surpassed classical models**, with an accuracy of **82.96%** and F1-score of **83.27%**.The **BERT model achieved the highest performance on the combined dataset**, with an accuracy of **90.49%**, precision of **91.51%**, recall of **89.23%**, and F1-score of **90.36%**. This highlights its capability to generalize effectively across diverse data sources and maintain robust classification performance.

5. VISUALIZATION OF RESULTS

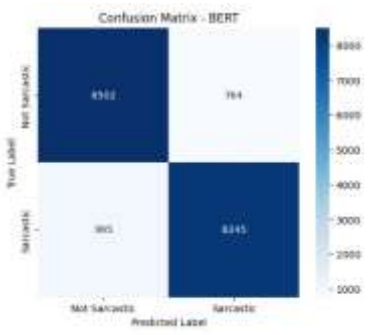

Figure 5: Confusion Matrix of BERT

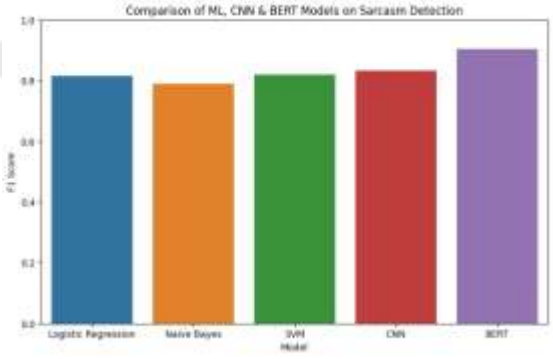

Figure 6: Comparison of ML, CNN & BERT models

Figure - 5 shows the confusion matrix of the **BERT model**, which achieved the best performance across datasets. Out of the total predictions, the model correctly classified **8,502 non-sarcastic samples** and **8,245 sarcastic samples**. Only **764 non-sarcastic samples** were misclassified as sarcastic, and **995 sarcastic samples** were misclassified as non-sarcastic. This indicates that **BERT**

is highly effective in distinguishing sarcasm, with relatively low misclassification rates compared to other models. The balanced diagonal distribution in the matrix highlights its robustness across both sarcastic and non-sarcastic categories.

Figure 6 presents a bar chart comparing the **F1-scores** of all models (Logistic Regression, Naïve Bayes, SVM, CNN, and BERT). It can be observed that:

- **Naïve Bayes achieved the lowest F1-score (0.79)**, indicating limited performance in handling sarcasm detection.
- **Logistic Regression and SVM performed similarly (0.81–0.82)**, showcasing moderate effectiveness.
- **CNN outperformed traditional ML models (0.83 F1-score)**, demonstrating the advantage of deep learning architectures.
- **BERT achieved the highest F1-score (0.90)**, confirming its superiority due to its contextual word representation and transformer-based learning.

6. CONCLUSION

The development and implementation of the hybrid sarcasm detection system have successfully demonstrated the feasibility and effectiveness of combining multiple NLP approaches — traditional machine learning (TF-IDF with Logistic Regression, Naive Bayes, SVM), deep learning (CNN), and transformer-based models (BERT) — into a unified framework. Through the integration of three diverse datasets (Kaggle headlines, extended headlines, and a conversational tweet dataset), the system achieved high accuracy in detecting sarcasm across different linguistic styles, sentence lengths, and content domains. Rigorous preprocessing, balanced training data, and optimized model parameters contributed to robust performance and resilience against noise in the input data. The system's architecture and modular design ensure maintainability, scalability, and ease of integration into web-based applications using tools like Streamlit. Testing confirmed that it meets the functional, performance, and usability requirements, while visualization features (confusion matrices, word clouds) provide interpretability for both researchers and end-users.Further this can be implemented in following things.

1. **Multilingual Support** – Extend sarcasm detection to multiple languages using multilingual transformers like XLM-RoBERTa or BERT, enabling a broader user base.
2. **Context-Aware Detection** – Incorporate conversational history or multi-turn dialogue context to improve accuracy in chat-based sarcasm detection.
3. **Sentiment-Sarcasm Fusion** – Combine sarcasm detection with sentiment analysis to provide deeper emotional and intent-based insights.
4. **Real-Time Social Media Monitoring** – Deploy the system as a live API for detecting sarcasm in real-time streams such as Twitter feeds or YouTube comments.
5. **Model Optimization for Edge Devices** – Optimize model size and inference time for deployment on mobile and low-resource devices.
6. **Explainable AI (XAI) Integration** – Integrate attention visualization and interpretability tools to explain why a sentence is classified as sarcastic.
7. **Custom Dataset Expansion** – Continuously update and expand the dataset with evolving slang, cultural references, and trending expressions.

REFERENCES

[1] Sinha, S., & Choudhary, M. (2023). Sarcasm Detection Using Deep Learning Approaches: A Review.Journal of Intelligent & Fuzzy Systems,

[2] Sarcasm Detection Using Deep Learning With Contextual Features. Razali, M., Doraisamy, S., Abdul Halin, A., Ye, L., & Norowi, N. M. (2022). Sarcasm Detection Using Deep Learning With Contextual Features.

[3] Sarcasm Detection: A Comparative Study. Yaghoobian, H., Arabnia, H. R., & Rasheed, K. (2021). Sarcasm Detection: A Comparative Study. In Proceedings of the 2021 International Conference on Artificial Intelligence (ICAI), The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp).

[4] Sarcasm Detection in Natural Language Processing. Aa, A., G, S., H R, S., Upadhyaya, M., Ray, A. P., & Manjunath, T. C. (2022).Sarcasm Detection in Natural Language Processing.

[5] Sarcasm Detection in Online Comments Using Machine Learning. Šandor, D., & Bagić Babac, M. (2021). Sarcasm Detection in Online Comments Using Machine Learning. In Proceedings of the 44th International Convention on Information, Communication and Electronic Technology (MIPRO), Opatija, Croatia. IEEE.

[6] A Deep Learning Approach for Sarcasm Detection in User-Generated Content. Chowdary, E. D., Sudheer, B. N., Sri, K. S., & Madhavi, P. R. (2023). A Deep Learning Approach for Sarcasm Detection in User-Generated Content. Procedia Computer Science, 218, 1710–1718.

[7] Kumar, A., & Gupta, R. (2023). Hybrid Deep Learning Models for Sarcasm Detection on Reddit.

[8] Hazarika, D. et al. (2018). CASCADE: Contextual Sarcasm Detection in Online Discussion Forums.In Proceedings of ACL 2018.

[9] Khandelwal, S., & Bedi, P. (2021). Multi-modal Sarcasm Detection Using Audio and Text Features.In Journal of Intelligent & Fuzzy Systems.

[10] Oprea, S., & Magdy, W. (2019). Exploring the Role of Context and User Embedding for Sarcasm Detection.
[11] Kumar, R., & Ramakrishnan, G. (2021). Ensemble learning for sarcasm detection using emotion cues.In Expert Systems with Applications.

[12] Vempala, A., & Preotiuc-Pietro, D. (2020). Categorizing and Inferring the Relationship between the Text and Image of Twitter Posts.

[13] Castro, S., & Lupu, M. (2021). MUStARD: A Multimodal Dataset for Sarcasm Detection in Dialogues.

[14] Bamman, D., Smith, N. A., & Eisenstein, J. (2020). Contextualized Sarcasm Detection on Reddit Using SARC Dataset.

[15] Agrawal, S., & Annamalai, S. (2023). Transformer-based Sarcasm Detection Using Tweet Embeddings.In Procedia Computer Science, 212.

[16] Naskar, S., & Dey, A. (2022). Capsule Networks for Sarcasm Detection.In Knowledge-Based Systems, 235.

[17] Mahajan, A., & Patel, P. (2022). Comparative Study of Transformer Architectures for Sarcasm Classification.In Neural Computing and Applications.

[18] Chopra, V., & Bansal, S. (2023). Fuzzy Logic Based Approach for Sarcasm and Irony Detection in Hindi-English Code-Mixed Data.Springer LNCS

[19] Nasir, M. et al. (2021). A Survey on Sarcasm Detection Using Machine Learning and Deep Learning.In Computers, Materials & Continua.

[20] Zhang, Y., & Liu, J. (2021). Adversarial Training for Sarcasm Detection Using BERT.