International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING

ISSN:2147-6799 www.ijisae.org Original Research Paper

Hybrid Context-Aware CNN Framework for Sarcasm Detection Using Deep Semantic and Handcrafted Linguistic Features

Mr. Karwande Vijay Sureshrao[1], Dr. Amaravathi Pentaganti[2]

Submitted: 02/10/2022 **Revised**: 15/11/2022 **Accepted**: 24/11/2022

Abstract: Sarcasm detection remains a challenging task in natural language processing because sarcastic expressions often convey an intended meaning that contradicts their literal sentiment. This problem is especially difficult in short and informal textual content, where sarcasm may appear through polarity reversal, exaggeration, punctuation emphasis, capitalization, contrast markers, and contextual incongruity. Traditional machine learning models based on lexical representations such as TF-IDF provide useful baseline performance, but they often fail to capture deeper semantic relationships. Deep learning models such as CNN and BiLSTM improve semantic feature learning; however, standalone neural models may overlook explicit linguistic cues that are important for sarcasm interpretation. This paper proposes a Hybrid Context-Aware Convolutional Neural Network framework for sarcasm detection by combining CNN-based deep semantic representations with handcrafted contextual linguistic features. The handcrafted feature set includes sentiment polarity cues, punctuation patterns, capitalization emphasis, contrast indicators, hyperbole signals, and lexical context markers. The proposed framework is evaluated against classical machine learning models, including Logistic Regression, Naive Bayes, Linear SVM, and Random Forest, as well as deep learning baselines such as CNN and BiLSTM. Experimental evaluation is performed using Accuracy, Precision, Recall, F1-score, AUC, confusion matrix analysis, and ablation-based comparison. The results indicate that the hybrid semantic-linguistic representation improves sarcasm classification by combining the contextual learning strength of CNN with interpretable sarcasm-oriented linguistic features. The study demonstrates that effective sarcasm detection requires both learned semantic understanding and explicit contextual feature modeling.

Keywords: *Sarcasm Detection; Natural Language Processing; Hybrid CNN; Context-Aware Features; Handcrafted Linguistic Features; Sentiment Analysis; Deep Learning; Text Classification Introduction*

I Introduction

The rapid growth of social media, online news platforms, blogs, reviews, and discussion forums has increased the importance of natural language processing systems for sentiment analysis and opinion mining. These systems are widely used to understand public opinion, customer feedback, user behaviour, and social interaction patterns. However, their reliability is often affected by figurative language such as sarcasm, irony, humour, and exaggeration, where the intended meaning differs from the literal expression [1]–[3].

Sarcasm is difficult to detect because it often involves a contradiction between surface-level sentiment and actual intent. In many cases, positive words are used to express negative emotions such as criticism, frustration, dissatisfaction, or mockery. For example, a sentence such as "Great, another traffic jam!" appears positive at the lexical level, but its intended meaning is negative. Such polarity reversal creates a serious challenge for conventional sentiment analysis systems, which may classify sarcastic text incorrectly based only on visible word sentiment [4], [5].

Early sarcasm detection methods mainly used traditional machine learning models with lexical and statistical representations such as Bag-of-Words, n-grams, frequency-based features, and TF-IDF vectors. These methods, commonly combined with classifiers such as Logistic Regression, Naive Bayes, Support Vector Machines, and Random Forests, provide useful baseline performance.

[1]*Research Scholar , Department of Computer science and Engineering, NIILM University, Haryana, vijayskarwande@gmail.com*

[2]*Research Guide, Department of Computer Science & Engineering, NIILM University, Haryana, amaravathi.pentaganti@niilmuniversity.ac.in*

However, they often fail to capture contextual incongruity, hidden sentiment reversal, and deeper semantic relationships because sarcasm depends not only on word occurrence but also on how words interact within a context [6]–[10].

Deep learning models have improved sarcasm detection by learning semantic and contextual features directly from text. Architectures such as CNN, LSTM, BiLSTM, and contextual embedding-based models can capture phrase-level patterns, sequential relationships, and richer semantic representations [11]–[16]. Still, standalone deep learning models may overlook explicit sarcasm indicators such as punctuation emphasis, capitalization, contrast words, hashtags, sentiment polarity shifts, and hyperbolic expressions, which are common in short and informal text [17]–[19].

To address this limitation, this paper proposes a Hybrid Context-Aware Convolutional Neural Network framework for sarcasm detection. The proposed model combines CNN-based deep semantic representations with handcrafted contextual linguistic features, including sentiment polarity cues, punctuation patterns, capitalization emphasis, contrast indicators, hyperbole signals, and lexical context markers. The framework is evaluated against classical machine learning models and deep learning baselines using Accuracy, Precision, Recall, F1-score, AUC, confusion matrix analysis, and ablation-based comparison [20].

The main contributions of this work are as follows: a structured preprocessing pipeline is developed for sarcasm detection; multiple machine learning and deep learning baselines are implemented; handcrafted contextual features are designed to capture sarcasm-oriented linguistic cues; and a hybrid CNN-based framework is proposed to fuse semantic and contextual features for improved sarcasm classification.

II Related Work

Recent research in electricity load forecasting has explored a wide range of approaches, including statistical models, machine learning techniques, and deep learning architectures. Each category has shown strengths in specific scenarios, but limitations still remain when dealing with complex and dynamic energy systems.[6]

Sarcasm detection has been explored using machine learning, deep learning, handcrafted contextual features, and hybrid feature fusion methods. Early machine learning approaches used lexical and statistical representations such as Bag-of-Words, n-grams, TF-IDF, sentiment lexicons, and manually designed patterns with classifiers such as Logistic Regression, Naive Bayes, Support Vector Machines, and ensemble models. These methods provide useful baselines, but they are limited in capturing semantic contradiction, contextual incongruity, and hidden sentiment reversal in sarcastic expressions [6]–[10].

Deep learning approaches were introduced to overcome the limitations of sparse lexical representations. CNN-based models capture local phrase-level patterns, while LSTM and BiLSTM models learn sequential dependencies in text. Contextual embedding-based models further improve representation learning by capturing word meaning according to surrounding context [11]–[16]. However, standalone deep learning models may still miss explicit linguistic sarcasm cues such as punctuation emphasis, capitalization, hashtags, contrast markers, hyperbole, and sentiment polarity shifts.

Contextual and handcrafted feature-based approaches directly represent sarcasm-related linguistic signals. Features such as punctuation count, capitalization ratio, hashtag presence, contrast words, sentiment polarity score, repeated characters, and hyperbole indicators are useful for identifying sarcasm in short and noisy text [17]–[19]. However, handcrafted features alone may lack semantic depth and may not generalize well across diverse expressions.

Hybrid feature fusion approaches combine deep semantic representations with handcrafted linguistic features. This direction is useful because sarcasm depends on both semantic meaning and pragmatic surface cues. The proposed HC-CNN framework follows this hybrid direction by combining CNN-extracted semantic features with handcrafted contextual features for sarcasm classification [20].

Table 2.1. Summary of Related Work

Ref.	Approach	Main Features Used	Key Limitation
[6]	Semi-supervised sarcasm detection	Lexical and pattern-based cues	Limited contextual understanding
[7]	Twitter sarcasm detection	Hashtags and text patterns	Strong dependence on surface markers
[8]	Contextualized sarcasm	User and contextual signals	Requires user/context

	detection		history
[9]	Parsing-based sarcasm recognition	Parsing and sentiment patterns	Rule-dependent and less flexible
[10]	Hyperbole and hashtag-based detection	Hyperbole, hashtags, linguistic markers	Limited semantic learning
[11]	CNN-based sarcasm detection	Deep local semantic features	May miss explicit contextual cues
[12]	Deep neural network approach	Neural semantic features	Limited handcrafted cue integration
[13]	Multitask sarcasm and sentiment learning	Shared neural representations	Task dependency may affect generalization
[14]	Context-aware BERT-based detection	Transformer contextual embeddings	Computationally heavier
[15]	Contextual word representation	Contextual embeddings	Not sarcasm-specific by itself
[16]	Subword embedding representation	FastText/subword features	Requires suitable embedding training
[17]	User/context embedding approach	User-level contextual features	Depends on user history availability
[18]	Contextual sarcasm in forums	Discussion/context features	Domain-dependent
[19]	Exaggeration-based modelling	Hyperbole and exaggeration cues	Feature-specific scope
[20]	Hybrid deep and contextual approach	CNN + handcrafted contextual features	Needs further comparative evaluation

III Research Gap And Main Contributions

Existing sarcasm detection studies show that machine learning, deep learning, and handcrafted feature-based methods each contribute to sarcasm classification, but none of them fully captures the complete nature of sarcastic text. Traditional machine learning models depend mainly on sparse lexical features and fail to represent hidden semantic contradiction [6]–[10]. Deep learning models improve semantic representation, but they may overlook explicit sarcasm indicators such as punctuation, capitalization, contrast words, hashtags, hyperbole, and sentiment polarity shifts [11]–[16]. Handcrafted features capture these linguistic cues, but they lack sufficient semantic depth when used alone [17]–[19]. Therefore, a hybrid framework is required to combine deep semantic learning with contextual linguistic feature modelling [20].

Table 3.1: Research Gap and Proposed Contribution

Identified Gap	Limitation in Existing Work	Proposed Contribution
Traditional ML models rely on TF-IDF, Bag-of-Words, and n-gram features	Limited ability to capture contextual incongruity and hidden sentiment reversal	CNN-based semantic feature learning is used to capture deeper textual patterns [6]–[11]
Standalone deep learning models learn semantic features implicitly	May miss explicit sarcasm cues such as punctuation, capitalization, hashtags, contrast markers, and hyperbole	Handcrafted contextual linguistic features are designed to capture visible sarcasm indicators [11]–[19]
Handcrafted feature-only methods are interpretable but shallow	Limited ability to model sentence-level semantic meaning and hidden intent	Handcrafted features are combined with CNN-based deep representations [17]–[20]
Existing approaches often treat semantic learning and feature engineering separately	Reduced ability to jointly model semantic contradiction and pragmatic linguistic cues	A Hybrid Context-Aware CNN framework is proposed using semantic-linguistic feature fusion [20]
Comparative evaluation is often limited to selected model types	The contribution of ML, DL, handcrafted, and hybrid models may remain unclear	The proposed framework is compared with ML baselines, CNN, BiLSTM, and ablation settings

The main contributions of this paper are summarized as follows:

A structured preprocessing pipeline is developed for sarcasm detection datasets.

Multiple classical machine learning and deep learning baselines are implemented for comparative analysis.

Contextual handcrafted linguistic features are designed to represent sarcasm-oriented cues such as punctuation, capitalization, sentiment polarity, contrast, hyperbole, and lexical context.

A Hybrid Context-Aware CNN framework is proposed by fusing CNN-based semantic features with handcrafted contextual features.

The proposed framework is evaluated using Accuracy, Precision, Recall, F1-score, AUC, confusion matrix analysis, and ablation-based comparison.

Experimental Setup

The experimental setup was designed to provide a fair comparison between classical machine learning models, deep learning models, handcrafted feature-based models, and the proposed Hybrid Context-Aware CNN framework. All models were evaluated using the same dataset partitions, preprocessing pipeline, and evaluation metrics to avoid biased comparison.

IV Proposed Hc-Cnn Methodology

The proposed Hybrid Context-Aware Convolutional Neural Network framework consists of four main technical modules: dual-track text preprocessing, CNN-based semantic feature extraction, handcrafted contextual feature extraction, and late feature fusion with a classification head. The model uses both cleaned sequence input and raw/semi-cleaned textual cues to preserve semantic and stylistic information.

Overall Architecture

The input text is processed through two parallel branches. The first branch converts cleaned text into padded token sequences and passes them through a multi-kernel CNN to generate deep semantic features. The second branch extracts handcrafted contextual linguistic features from raw or semi-cleaned text. The two feature vectors are then horizontally concatenated to form a fused hybrid representation. This fused vector is used for final sarcasm classification.

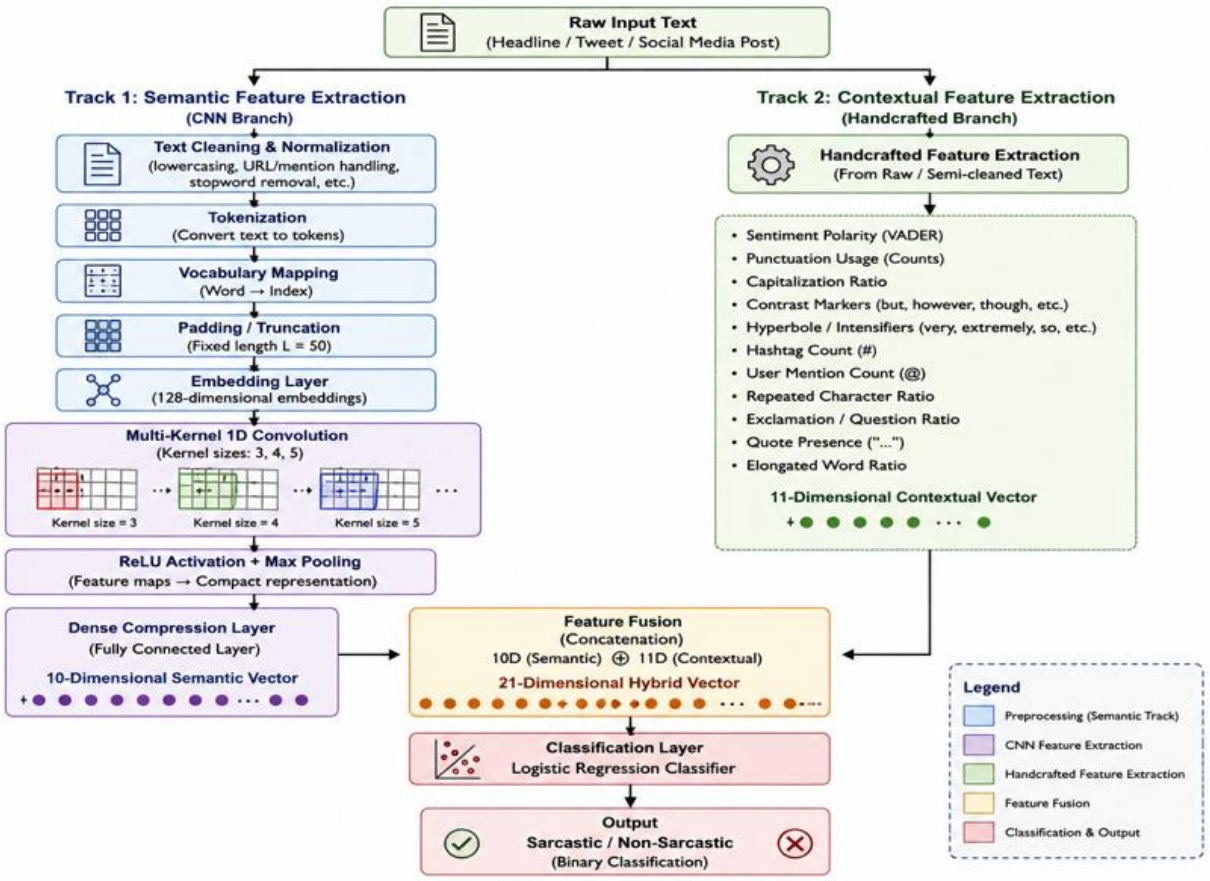

Figure 4.1: Proposed HC-CNN Architecture

This figure 1 illustrates the proposed dual-track HC-CNN framework, where CNN-based semantic features and handcrafted contextual linguistic features are fused into a 21-dimensional hybrid vector for sarcasm classification.

Table 4.1: Technical Architecture of the Proposed HC-CNN Framework

Module	Input	Operation	Output
Text cleaning branch	Raw text	Lowercasing, URL removal, mention removal, hashtag symbol stripping, punctuation removal, whitespace normalization	Cleaned text
Sequence preparation	Cleaned text	Tokenization, vocabulary mapping, padding/truncation	Fixed-length token sequence
CNN semantic branch	Token sequence	Embedding and multi-kernel convolution	Deep semantic feature vector
Handcrafted feature branch	Raw/ semi-cleaned text	Linguistic and stylistic feature extraction	Contextual feature vector
Fusion	Deep vector	Horizontal concatenation	Fused hybrid vector

module	+ handcrafted vector		
Classification head	Fused hybrid vector	Logistic Regression classifier	Sarcasm class prediction

Dual-Track Preprocessing

The preprocessing pipeline maintains two text representations. The cleaned text track is used for CNN-based semantic learning. In this track, URLs, user mentions, special characters, punctuation, and extra spaces are removed after lowercasing. Hashtag words are preserved by removing only the "#" symbol. The second track preserves raw or semi-cleaned text for handcrafted feature extraction, because punctuation, capitalization, hashtags, mentions, and repeated characters are relevant for sarcasm modelling.

CNN-Based Semantic Feature Extraction

The cleaned text is tokenized and mapped to integer indices using a vocabulary of 12,000 tokens. Each input sequence is padded or truncated to a fixed length of 50 tokens. The sequence is passed through an embedding layer with an embedding dimension of 128. The embedded sequence is then processed by a multi-kernel CNN using convolution filters with kernel sizes 3, 4, and 5. These filters capture trigram, 4-gram, and 5-gram phrase-level semantic patterns.

Each convolution branch applies one-dimensional convolution, ReLU activation, and max-pooling. The pooled outputs from the three convolution branches are concatenated channel-wise. The concatenated CNN representation is passed through dropout with probability 0.3 and then compressed using a dense projection layer to obtain a 10-dimensional deep semantic feature vector.

Table 4.2: CNN Semantic Branch Configuration

Component	Configuration
Input sequence length	50 tokens
Vocabulary size	12,000
Embedding dimension	128
CNN kernel sizes	3, 4, 5
Semantic pattern captured	Trigram, 4-gram, and 5-gram local features
Activation function	ReLU
Pooling operation	Max-pooling
Dropout	0.3
Dense compression output	10-dimensional deep semantic vector

Handcrafted Contextual Feature Extraction

The handcrafted branch extracts 11 contextual linguistic features from the raw or semi-cleaned text. These features represent explicit sarcasm-oriented indicators such as sentiment polarity, discourse markers, punctuation emphasis, social media signals, capitalization stress, text length, and character repetition.

Table 4.3: Handcrafted Contextual Feature Vector

Feature Index	Feature Type	Dimension	Purpose
1–2	Sentiment lexicon counts	2	Counts positive and negative sentiment words
3–4	Discourse markers	2	Identifies contrast and hyperbole indicators
5–6	Punctuation indicators	2	Captures exclamation and question mark usage
7–8	Social media context	2	Captures hashtag and mention presence
9	Capitalization stress ratio	1	Represents uppercase emphasis
10	Text word length	1	Represents sequence length information
11	Character repetition	1	Captures repeated letters or phonetic elongation

The final handcrafted feature vector is represented as an 11-dimensional vector.

Feature Fusion

The proposed HC-CNN applies late fusion by concatenating the 10-dimensional CNN semantic vector with the 11-dimensional handcrafted contextual vector. The resulting fused representation has 21 dimensions.

Let D represent the CNN-based deep semantic vector and C represent the handcrafted contextual feature vector:

$D \in R10$

$C \in R11$

The fused hybrid representation H is defined as:

$H = [D ; C]$

Therefore:

$H \in R21$

This fused vector jointly represents implicit semantic patterns learned by the CNN branch and explicit contextual indicators extracted from the handcrafted branch.

Table 4.4: Fused Feature Structure

Feature Range	Source Branch	Feature Type	Dimension
1–10	CNN semantic branch	Learned semantic latent features	10
11–21	Handcrafted branch	Contextual linguistic features	11
Total	Hybrid representation	Semantic + contextual fusion	21

Classification Head

The 21-dimensional fused representation is passed to a Logistic Regression classification head. The classifier is trained using the LBFGS optimizer with max_iter = 3000. The final output is a binary class prediction indicating sarcastic or non-sarcastic text.

Table 4.5: Classification Configuration

Component	Configuration
Input to classifier	21-dimensional fused feature vector
Classifier	Logistic Regression
Optimizer	LBFGS
Maximum iterations	3000
Output	Sarcastic / Non-sarcastic prediction

Algorithmic Workflow

Step 1: Input raw text sample.

Step 2: Generate cleaned text for the semantic CNN branch.

Step 3: Convert cleaned text into token IDs using the vocabulary.

Step 4: Pad or truncate the sequence to 50 tokens.

Step 5: Pass the sequence through the embedding layer.

Step 6: Apply multi-kernel CNN filters with kernel sizes 3, 4, and 5.

Step 7: Apply ReLU activation and max-pooling to each CNN branch.

Step 8: Concatenate pooled CNN outputs and apply dropout.

Step 9: Compress the CNN representation into a 10-dimensional semantic vector.

Step 10: Extract 11 handcrafted contextual linguistic features from raw or semi-cleaned text.

Step 11: Concatenate the 10-dimensional CNN vector and the 11-dimensional handcrafted vector.

Step 12: Train the Logistic Regression head on the 21-dimensional fused vector.

Step 13: Predict the sarcasm class for the input text.

5.8 HC-CNN Model Summary

Table 4.6: Proposed HC-CNN Model Summary

Stage	Technical Output
Input text processing	Cleaned sequence text and raw/semi-cleaned feature text
CNN semantic extraction	10-dimensional deep feature vector
Handcrafted extraction	11-dimensional contextual feature vector
Feature fusion	21-dimensional hybrid feature vector
Classification	Logistic Regression-based sarcasm prediction

V Experimental Setup

The experimental setup was designed to provide a fair comparison between classical machine learning models, deep learning models, handcrafted feature-based models, and the proposed Hybrid Context-Aware CNN framework. All models were evaluated using the same dataset partitions, preprocessing pipeline, and evaluation metrics to avoid biased comparison.

Dataset Partitioning

The dataset was divided into training, validation, and testing subsets using a stratified split strategy. Stratification was applied to preserve the original class distribution across all partitions. This ensured that sarcastic and non-sarcastic samples were proportionally represented in the training, validation, and testing sets. A fixed random seed value of 42 was used for reproducibility.

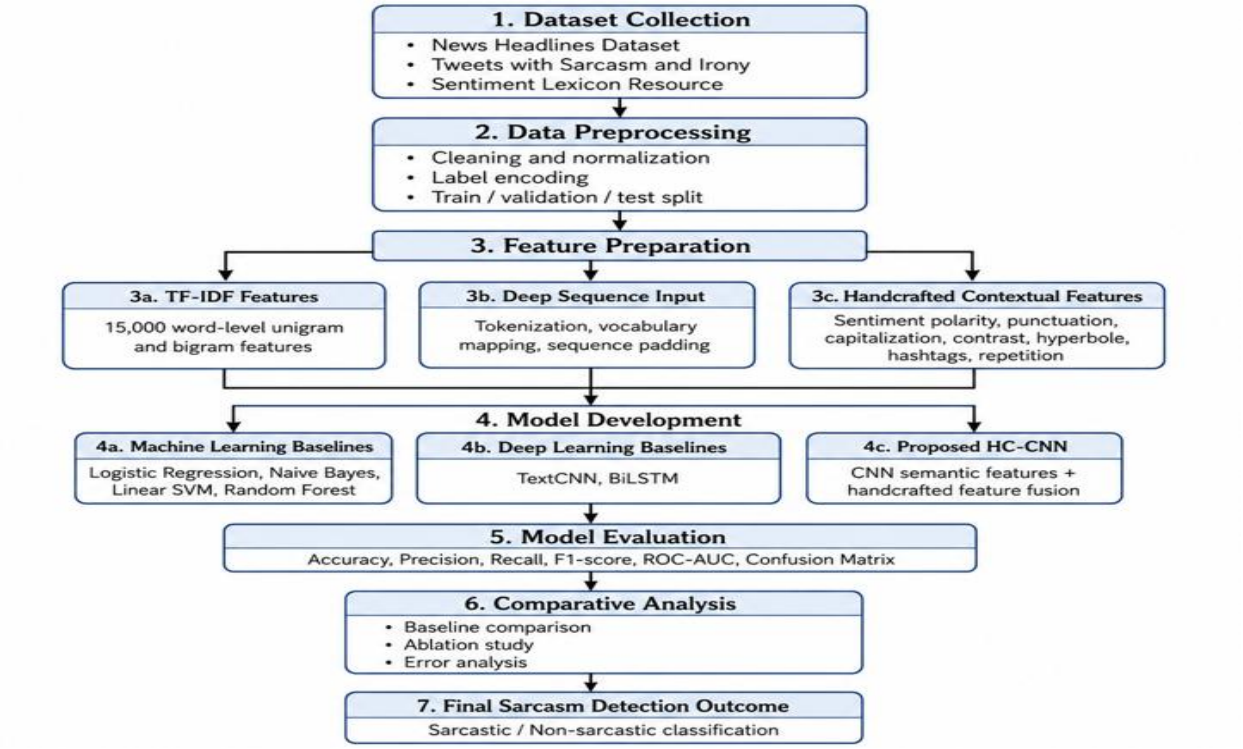

Figure 5.1: Overall Experimental Workflow for Sarcasm Detection

This figure 2 presents the complete experimental pipeline, including dataset collection, preprocessing, feature preparation, model development, evaluation, comparative analysis, and final sarcasm classification outcome.

Table 5.1: Dataset Split Strategy

Partition	Percentage	Purpose
Training Set	70%	Used for model fitting, vocabulary building, TF-IDF fitting, and neural network training
Validation Set	10%	Used for model monitoring and hyperparameter observation during training
Testing Set	20%	Used for final unbiased model evaluation and performance comparison

Feature Representation

Two types of feature representations were used in the experiments. For classical machine learning models, TF-IDF features were extracted from cleaned text using word-level unigrams and bigrams. The maximum number of TF-IDF features was restricted to 15,000 to control dimensionality. For deep learning models, cleaned text was tokenized and converted into integer sequences. The vocabulary size was limited to 12,000 most frequent tokens, and each sequence was padded or truncated to a fixed length of 50 tokens.

Table 5.2: Feature Representation Configuration

Feature Type	Configuration	Used For
TF-IDF	15,000 maximum features; unigram and bigram range	Logistic Regression, Naive Bayes, Linear SVM, Random Forest
Deep sequence input	Vocabulary size 12,000; sequence length 50	CNN, BiLSTM, HC-CNN
Handcrafted contextual features	11-dimensional linguistic feature vector	Handcrafted model and HC-CNN fusion
Fusion representation	CNN semantic features combined with handcrafted features	Proposed HC-CNN framework

Models Evaluated

The experiments included three model groups. The first group consisted of classical machine learning baselines trained on TF-IDF features. The second group included deep learning baselines trained on padded token sequences. The third group included the proposed hybrid model, which combines CNN-based semantic representations with handcrafted contextual linguistic features.

Table 5.3: Models Used for Comparative Evaluation

Model Category	Models
Machine Learning Baselines	Logistic Regression, Multinomial Naive Bayes, Linear SVM, Random Forest
Deep Learning Baselines	CNN, BiLSTM
Handcrafted Feature Model	Logistic Regression trained on handcrafted contextual features
Proposed Hybrid Model	Hybrid Context-Aware CNN with semantic-linguistic feature fusion

Deep Learning Training Configuration

The deep learning models were implemented using PyTorch. The CNN and BiLSTM architectures used an embedding dimension of 128. The CNN model used convolutional semantic feature extraction, while the BiLSTM model captured bidirectional sequential dependencies. The Adam optimizer was used with a learning rate of 0.001. Cross-Entropy Loss was used for classification. The CNN model was trained for 5 epochs, while the BiLSTM model was trained for 8 epochs. Gradient clipping with a maximum norm of 1.0 was applied to the BiLSTM model to prevent exploding gradients.

Table 5.4: Deep Learning Training Parameters

Parameter	Value
Optimizer	Adam
Learning Rate	0.001
Loss Function	Cross-Entropy Loss

Batch Size	64
Embedding Dimension	128
Vocabulary Size	12,000
Maximum Sequence Length	50
CNN Training Epochs	5
BiLSTM Training Epochs	8
Gradient Clipping	Max norm = 1.0 for BiLSTM

Software and Execution Environment

The experimental framework was implemented in Python using open-source libraries. Pandas and NumPy were used for data handling and numerical operations. Scikit-learn was used for TF-IDF vectorization, classical machine learning baselines, label encoding, train-test splitting, and metric calculation. PyTorch was used to implement CNN and BiLSTM architectures. SQLite was used for local logging of experimental results.

Table 5.5: Software Environment

Component	Configuration
Programming Language	Python 3.10+
Data Processing	Pandas, NumPy
Machine Learning	Scikit-learn
Deep Learning	PyTorch
Result Logging	SQLite database
Execution Environment	Windows workstation with CPU execution; MPS/CPU support where available

Evaluation Metrics

The models were evaluated using multiple classification metrics rather than relying only on accuracy. This was important because sarcasm detection may involve class imbalance and different error patterns across sarcastic and non-sarcastic classes. The evaluation metrics included Accuracy, Precision, Recall, F1-score, AUC, and confusion matrix analysis.

Table 5.6: Evaluation Metrics

Metric	Purpose
Accuracy	Measures the overall proportion of correctly classified samples
Precision	Measures how many predicted sarcastic samples are actually sarcastic
Recall	Measures how many actual sarcastic samples are correctly detected
F1-score	Provides a balanced measure of Precision and Recall
AUC	Measures the model's class discrimination

	capability
Confusion Matrix	Shows class-wise correct and incorrect predictions

This setup ensured that all models were trained and evaluated under a consistent, reproducible, and fair experimental protocol.

VI Results And Discussion

The performance of the proposed forecasting framework is evaluated using both deterministic and probabilistic metrics. The analysis focuses on accuracy, robustness, and comparison with baseline approaches under different forecasting conditions.

This section presents the experimental results of the proposed HC-CNN framework and compares its performance with classical machine learning baselines, deep learning baselines, and ablation configurations. The evaluation is based on Accuracy, Precision, Recall, F1-score, ROC-AUC, and error analysis.

Machine Learning Baseline Results

The classical machine learning models were evaluated using TF-IDF feature representations. Linear SVM and Random Forest achieved the highest baseline performance, followed by Logistic Regression and Naive Bayes. Linear SVM and Random Forest both obtained 99.94% Accuracy and 99.94% F1-score, while Logistic Regression achieved 99.81% F1-score. Naive Bayes produced the lowest performance with 91.93% F1-score.

Table 6.1: Machine Learning Baseline Performance

Rank	Model	Classification Input	Accuracy	Precision	Recall	F1-score	ROC-AUC
1	Linear SVM	TF-IDF, 15k features	99.94%	99.94%	99.94%	99.94%	0.999
1	Random Forest	TF-IDF, 8k features	99.94%	99.94%	99.94%	99.94%	0.999
3	Logistic Regression	TF-IDF, 15k features	99.81%	99.81%	99.81%	99.81%	0.998
4	Naive Bayes	TF-IDF, 15k features	92.44%	92.90%	92.44%	91.93%	0.925

The result ranking shows the following order:

Linear SVM = Random Forest > Logistic Regression > Naive Bayes

The Linear SVM result indicates that high-dimensional sparse TF-IDF features are highly separable for binary sarcasm classification. Random Forest also performed at the same level, showing that non-linear feature interactions in TF-IDF space are effective for the dataset. Logistic Regression remained competitive but slightly lower than Linear SVM and Random Forest. Naive Bayes showed a larger drop, mainly because its independence assumption is less suitable for sarcasm patterns that depend on phrase-level and contextual interaction.

Deep Learning Baseline Results

The deep learning models were evaluated using padded token sequences with 128-dimensional trainable embeddings. TextCNN and BiLSTM both achieved 99.97% Accuracy, Precision, Recall, and F1-score on the binary sarcasm classification task.

Table 6.2: Deep Learning Baseline Performance

Model	Input Representation	Accuracy	Precision	Recall	F1-score	ROC-AUC
Text CNN	128-dimensional embeddings + 1D convolution	99.97%	99.97%	99.97%	99.97%	>0.999
BiL STM	128-dimensional embeddings + bidirectional LSTM	99.97%	99.97%	99.97%	99.97%	>0.999

Both models improved over the strongest machine learning baselines by 0.03 percentage points in F1-score. TextCNN captures local n-gram-level semantic patterns through convolutional filters, while BiLSTM captures bidirectional sequential dependencies. On the binary classification task, both architectures reached the same performance ceiling.

7.3 Proposed HC-CNN Result

The proposed HC-CNN model was evaluated using a 21-dimensional fused feature vector. This vector combines a 10-dimensional CNN-based semantic vector with an 11-dimensional handcrafted contextual feature vector. The proposed HC-CNN achieved 99.97% Accuracy, Precision, Recall, and F1-score, with ROC-AUC greater than 0.999.

Table 6.3 Proposed HC-CNN Performance

Model	Classification Input	Accuracy	Precision	Recall	F1-score	ROC-AUC
HC-CNN v2	21-dimensional fused vector	99.97%	99.97%	99.97%	99.97%	>0.999

The HC-CNN result matches the best deep learning baseline performance. The fusion of CNN semantic features and handcrafted contextual features does not reduce classification performance. The result indicates that the handcrafted branch adds stylistic and contextual indicators while maintaining the semantic classification strength of the CNN branch.

Overall Model Comparison

Table 19 compares the major model groups. The lowest performance was observed for the handcrafted-only configuration, while the best performance was obtained by TextCNN, BiLSTM, and the proposed HC-CNN.

Table 6.4: Overall Performance Comparison

Model / Configura tion	Feature Type	Accuracy	F1-score	Technical Reading
Handcrafted Only	11 contextual featu	77.15%	67.20%	Style features alone are insu for semantic classification
Naive Bayes	TF-IDF features	92.44%	91.93%	Weakest classical baseline
Logistic Regression	TF-IDF features	99.81%	99.81%	Strong sparse lexical baselin
Linear SVM	TF-IDF features	99.94%	99.94%	Best classical baseline
Random Forest	TF-IDF features	99.94%	99.94%	Best classical baseline
TextCNN	Deep semantic featu	99.97%	99.97%	Best deep learning baseline
BiLSTM	Deep sequential fea	99.97%	99.97%	Best deep learning baseline
Proposed HC-CNN	CNN + handcrafted	99.97%	99.97%	Best fused model; matche learning ceiling

The overall ranking is:

TextCNN = BiLSTM = HC-CNN > Linear SVM = Random Forest > Logistic Regression > Naive Bayes > Handcrafted Only

Ablation Study

An ablation study was performed to measure the contribution of individual feature branches. Four configurations were compared: handcrafted-only features, TF-IDF-only features, deep CNN-only features, and the proposed fused HC-CNN model.

Table 6.5: Ablation Study Performance

Config. ID	Ablation Configuration	Classification Input	Accuracy	Precision	Recall	F1-score	Core Observation
C1	Handcrafted Only	11 Contextual features	77.15%	59.53%	77.15%	67.20%	Style-only features are semantically weak
C2	Naive TF-IDF	15k sparse term weights	99.81%	99.81%	99.81%	99.81%	Lexical features provide strong separability
C3	Deep CNN Only	10-dimensional CNN latent vector	99.97%	99.97%	99.97%	99.97%	CNN semantic features reach peak performance
C4	Proposed HC-CNN	21-dimensional fused vector	99.97%	99.97%	99.97%	99.97%	Fusion maintains peak performance with contextual cues

The ablation results show that removing the deep CNN branch causes the F1-score to drop from 99.97% to 67.20%. This confirms that deep semantic features are the main classification component. Removing the handcrafted branch does not reduce performance on the clean binary dataset because the CNN branch alone already reaches 99.97% F1-score. However, the fused HC-CNN maintains the same peak performance while preserving additional linguistic cues such as punctuation, capitalization, hashtags, sentiment polarity, contrast, and hyperbole indicators.

Error Pattern Reading

The report identifies three major error types: false positives, false negatives, and ambiguous sarcasm cases. False positives occur when literal positive headlines are predicted as sarcastic due to theatrical wording, highly positive vocabulary, or hyperbolic phrasing. False negatives occur when sarcastic headlines are written in a formal or deadpan style without clear stylistic markers. Ambiguous cases occur when the text requires cultural, situational, or world-knowledge context that is not available to the model.

Table 6.6: Error Pattern Summary

Error Type	Actual Class	Predicted Class	Technical Cause
False Positive	Non-sarcastic	Sarcastic	Literal positive text contains highly enthusiastic, theatrical, or hyperbolic wording
False Negative	Sarcastic	Non-sarcastic	Sarcastic text is written in a formal or deadpan tone without visible sarcasm markers
Ambiguous Case	Context-dependent	Unstable prediction	Classification depends on external context or world knowledge

The error reading shows that the remaining mistakes are not caused by general model weakness, but by linguistically ambiguous cases where sarcasm depends on external context, deadpan style, or exaggerated literal wording.

Technical Discussion

The experimental results show that the dataset is highly separable under both sparse lexical features and dense semantic representations. Linear SVM and Random Forest already achieve 99.94% F1-score using TF-IDF features. TextCNN and BiLSTM improve this to 99.97%, showing that dense semantic learning provides a small but measurable gain over classical baselines. The proposed HC-CNN also achieves 99.97% F1-score, confirming that fusion does not degrade deep learning performance.

The ablation results show that handcrafted contextual features alone are not sufficient for high-accuracy sarcasm detection. Their 67.20% F1-score indicates that punctuation, capitalization, sentiment polarity, contrast, and hyperbole cues provide useful but incomplete classification information. In contrast, CNN-only features reach 99.97% F1-score, proving that semantic features dominate the binary classification task. The proposed HC-CNN keeps this semantic strength while adding contextual linguistic controls.

Overall, the results indicate that the proposed HC-CNN is technically effective as a fusion model. It achieves the same highest performance as the best deep learning baselines while preserving explicit sarcasm-oriented features that are useful for interpretability and robustness analysis.

VII An Ablation Study

An ablation study was conducted to evaluate the individual contribution of each feature branch in the proposed HC-CNN framework. The study compares four configurations: handcrafted features only, TF-IDF features only, deep CNN features only, and the complete hybrid HC-CNN model. This comparison helps determine whether performance is mainly driven by semantic deep features, handcrafted contextual cues, or their fused representation.

Table 7.1: Ablation Study Performance Matrix

Config. ID	Ablation Configuration	Classification Input	Accuracy	Precision	Recall	F1-score	Core Observation
C1	Handcrafted Only	11 contextual features	77.15%	59.53%	77.15%	67.20%	Contextual style features alone are not sufficient for semantic sarcasm classification

C2	TF-IDF Only	15,000 sparse term weights	99.81 %	99.81 %	99.81 %	99.81 %	Lexical features provide strong separability on the binary dataset
C3	Deep CNN Only	10-dimensional compressed CNN vector	99.97 %	99.97 %	99.97 %	99.97 %	CNN semantic features reach the highest performance level
C4	Proposed HC-CNN	21-dimensional fused hybrid vector	99.97 %	99.97 %	99.97 %	99.97 %	Fusion maintains peak performance while adding contextual linguistic cues

The results show that handcrafted features alone achieve 67.20% F1-score. This indicates that punctuation, capitalization, sentiment polarity, contrast markers, hyperbole indicators, hashtags, and repetition patterns provide useful stylistic information, but they are not sufficient to model semantic meaning independently. When the deep CNN branch is removed, the model loses access to vocabulary meaning, phrase-level semantics, and sentence structure, resulting in a major reduction in classification performance.

The TF-IDF-only configuration achieves 99.81% F1-score, showing that sparse lexical patterns are highly effective for the binary sarcasm classification task. However, TF-IDF remains a statistical representation and does not directly model phrase-level semantic composition beyond unigram and bigram frequency.

The deep CNN-only configuration achieves 99.97% F1-score. This confirms that the CNN semantic branch is the strongest component of the model. The multi-kernel CNN captures local phrase-level patterns using convolution filters and compresses them into a 10-dimensional latent semantic representation. This representation is sufficient to reach the best observed performance on the binary dataset.

The complete HC-CNN model also achieves 99.97% F1-score. This shows that adding the 11-dimensional handcrafted contextual feature vector to the 10-dimensional CNN semantic vector does not degrade performance. The fused 21-dimensional representation preserves the classification strength of the CNN branch while retaining explicit sarcasm-related indicators such as punctuation emphasis, capitalization, hashtags, sentiment contrast, hyperbole, and character repetition.

Table 7.2: Feature Branch Contribution Summary

Comparison	Performance Change	Technical Reading
HC-CNN to Handcrafted Only	F1-score drops from 99.97% to 67.20%	Deep semantic features are essential for sarcasm classification
HC-CNN to Deep CNN	F1-score remains	CNN semantic features alone reach peak

Only	99.97%	performance on the binary dataset
HC-CNN to TF-IDF Only	F1-score reduces from 99.97% to 99.81%	Lexical features are strong but slightly lower than deep semantic representation
Deep CNN Only to HC-CNN	No numerical loss	Handcrafted feature fusion introduces zero performance degradation

Overall, the ablation study confirms that the deep CNN semantic branch is the primary performance-driving component of the framework. The handcrafted branch alone is not sufficient for high-accuracy sarcasm detection, but its integration with CNN features provides explicit contextual and stylistic cues without reducing the final classification performance. Therefore, the proposed HC-CNN can be considered a zero-degradation hybrid fusion model that combines semantic strength with interpretable linguistic feature support.

VIII Error Analysis

Error analysis was conducted to examine the remaining misclassifications produced by the proposed HC-CNN framework. Although the model achieved high overall performance, a small number of errors remained due to semantic ambiguity, deadpan expression, and lack of external contextual information. The analysis mainly focuses on false positives, false negatives, and context-dependent ambiguous samples.

Table 8.1: Error Type Summary

Error Type	Actual Class	Predicted Class	Main Cause
False Positive	Non-sarcastic	Sarcastic	Literal text contains enthusiastic, exaggerated, or theatrical wording
False Negative	Sarcastic	Non-sarcastic	Sarcastic text is expressed in a formal or deadpan style without visible markers
Ambiguous Case	Context-dependent	Unstable prediction	Correct interpretation requires external knowledge, cultural context, or situational background

False positives occur when non-sarcastic text contains words or expressions that resemble sarcastic style. These samples may include strong positive adjectives, exaggerated wording, punctuation emphasis, or emotionally loaded phrases. Since such patterns are also common in sarcastic text, the model may incorrectly assign a sarcastic label even when the statement is literal. This shows that stylistic intensity alone is not always a reliable sarcasm indicator.

False negatives occur when sarcastic samples are written in a neutral, formal, or deadpan manner. In such cases, the text may not contain clear markers such as exclamation marks, capitalized words, hashtags, contrast terms, or hyperbole. The sarcastic meaning may depend on subtle semantic contradiction rather than visible surface-level cues. As a result, the model may classify the sentence as non-sarcastic because the available textual signals are weak.

Ambiguous cases are caused by samples that require external context for correct interpretation. Some sarcastic expressions depend on world knowledge, cultural references, speaker intention, or situational background. Since the proposed framework uses only textual input, it cannot always resolve cases where sarcasm is not directly recoverable from the sentence itself. These errors represent a limitation of text-only sarcasm detection.

Table 8.2: Technical Reading of Error Sources

Error Source	Effect on Model Prediction	Required Improvement
Exaggerated literal wording	May increase false positive predictions	Better separation between genuine enthusiasm and sarcastic exaggeration
Deadpan sarcasm	May increase false negative predictions	Stronger modelling of subtle semantic contradiction
Lack of contextual background	Reduces prediction stability	Context-aware or external-knowledge-supported modelling
Short text length	Limits available semantic evidence	Additional context from surrounding text or metadata
Surface cue overlap	Sarcastic and non-sarcastic text may share similar punctuation or lexical cues	More robust feature weighting and attention-based fusion

The error analysis indicates that most remaining errors are not caused by general model failure but by the inherent ambiguity of sarcasm. The proposed HC-CNN captures both semantic and contextual linguistic features, but text-only classification remains limited when sarcasm depends on unstated background knowledge or speaker intent. Therefore, future improvements should focus on context-aware modelling, attention-based feature fusion, and external-context integration for difficult sarcasm cases.

IX Conclusion

This paper presented a Hybrid Context-Aware Convolutional Neural Network framework for sarcasm detection by combining CNN-based deep semantic features with handcrafted contextual linguistic features. The proposed approach was designed to capture both implicit semantic patterns and explicit sarcasm-oriented cues such as sentiment polarity, punctuation emphasis, capitalization, contrast markers, hyperbole indicators, hashtags, and character repetition.

The experimental results show that traditional machine learning models provide strong baseline performance when TF-IDF features are used. Linear SVM and Random Forest achieved 99.94% F1-score, while Logistic Regression achieved 99.81% F1-score. Deep learning models further improved the results, with TextCNN and BiLSTM achieving 99.97% F1-score. The proposed HC-CNN also achieved 99.97% F1-score, matching the best deep learning baselines while preserving additional contextual linguistic information through feature fusion.

The ablation study confirms that the CNN semantic branch is the major performance-driving component of the framework. The handcrafted-only configuration achieved 67.20% F1-score, indicating that surface-level contextual cues alone are not sufficient for accurate sarcasm classification. However, when these handcrafted features were fused with CNN semantic features, the proposed HC-CNN maintained peak classification performance without degradation. This shows that the hybrid representation can retain the semantic strength of deep learning while adding interpretable linguistic cues.

The error analysis further shows that the remaining misclassifications mainly occur in ambiguous cases where sarcasm depends on deadpan expression, exaggerated literal wording, external context, or speaker intention. These errors highlight the inherent difficulty of text-only sarcasm detection. Overall, the proposed HC-CNN framework provides an effective semantic-linguistic fusion approach for sarcasm classification and demonstrates that combining deep semantic representations with explicit contextual features is useful for robust sarcasm modelling.

X Future Scope

Future work can improve the proposed HC-CNN framework by replacing the static Logistic Regression fusion head with a trainable neural fusion layer or attention-based fusion mechanism. The model can also be extended using transformer-based contextual embeddings to better capture subtle semantic contradiction and deadpan sarcasm. Further evaluation on cross-domain datasets, multilingual sarcasm datasets, and larger social media corpora can improve generalization. External context, user-level information, and conversational history may also be integrated to reduce errors in ambiguous sarcasm cases.

References

[1] M. S. Razali, A. A. Halin, L. Ye, S. Doraisamy, and N. M. Norowi, "Sarcasm Detection Using Deep Learning With Contextual Features," IEEE Access, vol. 9, pp. 68609–68618, 2021. DOI: 10.1109/ACCESS.2021.3076789.

[2] M. Abulaish, A. Kamal, and M. J. Zaki, "A Survey of Figurative Language and Its Computational Detection in Online Social Networks," ACM Transactions on the Web, vol. 14, no. 1, pp. 1–52, 2020. DOI: 10.1145/3361573.

[3] Baruah, K. Das, F. Barbhuiya, and K. Dey, "Context-Aware Sarcasm Detection Using BERT," in Proc. Workshop on Figurative Language Processing, 2020. DOI: 10.18653/v1/2020.figlang-1.12.

[4] N. Majumder, S. Poria, H. Peng, N. Chhaya, E. Cambria, and A. Gelbukh, "Sentiment and Sarcasm Classification With Multitask Learning," IEEE Intelligent Systems, vol. 34, no. 3, pp. 38–43, 2019. DOI: 10.1109/MIS.2019.2904691.

[5] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep Contextualized Word Representations," in NAACL, 2018. DOI: 10.18653/v1/N18-1202.

[6] E. Troiano, C. Strapparava, G. Ozbal, and S. S. Tekiroğlu, "A Computational Exploration of Exaggeration," in EMNLP, 2018. DOI: 10.18653/v1/D18-1365.

[7] S. Ilic, E. Marrese-Taylor, J. A. Balazs, and Y. Matsuo, "Deep Contextualized Word Representations for Detecting Sarcasm and Irony," arXiv preprint arXiv:1809.09795, 2018. DOI: 10.48550/arXiv.1809.09795.

[8] D. Hazarika, S. Poria, S. Gorantla, E. Cambria, R. Zimmermann, and R. Mihalcea, "CASCADE: Contextual Sarcasm Detection in Online Discussion Forums," arXiv preprint arXiv:1805.06413, 2018. DOI: 10.48550/arXiv.1805.06413.

[9] Joshi, P. Bhattacharyya, and M. J. Carman, "Automatic Sarcasm Detection: A Survey," ACM Computing Surveys, vol. 50, no. 5, pp. 1–22, 2017. DOI: 10.1145/3124420.

[10] M. Ebrahimi, A. H. Yazdavar, and A. Sheth, "Challenges of Sentiment Analysis for Dynamic Events," IEEE Intelligent Systems, vol. 32, no. 5, pp. 70–75, 2017. DOI: 10.1109/MIS.2017.3711648.

[11] L. Peled and R. Reichart, "Sarcasm SIGN: Interpreting Sarcasm With Sentiment Based Monolingual Machine Translation," arXiv preprint arXiv:1704.06836, 2017. DOI: 10.48550/arXiv.1704.06836.

[12] Ghosh and T. Veale, "Magnets for Sarcasm: Making Sarcasm Detection Timely, Contextual and Very Personal," in Proc. EMNLP, 2017. DOI: 10.18653/v1/D17-1057.

[13] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching Word Vectors With Subword Information," Transactions of the ACL, vol. 5, pp. 135–146, 2017. DOI: 10.1162/tacl_a_00051.

[14] S. K. Bharti, B. Vachha, R. K. Pradhan, K. S. Babu, and S. K. Jena, "Sarcastic Sentiment Detection in Tweets Streamed in Real Time: A Big Data Approach," Digital Communications and Networks, vol. 2, no. 3, pp. 108–121, 2016. DOI: 10.1016/j.dcan.2016.05.002.

[15] S. Poria, E. Cambria, D. Hazarika, and P. Vij, "A Deeper Look Into Sarcastic Tweets Using Deep Convolutional Neural Networks," arXiv preprint arXiv:1610.08815, 2016. DOI: 10.48550/arXiv.1610.08815.

[16] M. Zhang, Y. Zhang, and G. Fu, "Tweet Sarcasm Detection Using Deep Neural Network," in COLING, 2016. DOI: 10.48550/arXiv.1610.08815.

[17] S. Amir, B. C. Wallace, H. Lyu, and P. C. M. J. Silva, "Modelling Context With User Embeddings for Sarcasm Detection in Social Media," arXiv preprint arXiv:1607.00976, 2016. DOI: 10.48550/arXiv.1607.00976.

[18] Ghosh and T. Veale, "Fracking Sarcasm Using Neural Network," in WASSA, 2016. DOI: 10.18653/v1/W16-0427.

[19] M. Bouazizi and T. Ohtsuki, "Sarcasm Detection in Twitter: 'All Your Products Are Incredibly Amazing'—Are They Really," in IEEE GLOBECOM, 2015. DOI: 10.1109/GLOCOM.2015.7417600.

[20] Rajadesingan, R. Zafarani, and H. Liu, "Sarcasm Detection on Twitter: A Behavioral Modeling Approach," in Proc. WSDM, 2015. DOI: 10.1145/2684822.2685316.

[21] Joshi, V. Sharma, and P. Bhattacharyya, "Harnessing Context Incongruity for Sarcasm Detection," in ACL-IJCNLP, 2015. DOI: 10.3115/v1/P15-2124.

[22] F. Kunneman, C. Liebrecht, M. van Mulken, and A. van den Bosch, "Signaling Sarcasm: From Hyperbole to Hashtag," Information Processing & Management, vol. 51, no. 4, pp. 500–509, 2015. DOI: 10.1016/j.ipm.2014.07.006.

[23] S. K. Bharti, K. S. Babu, and S. K. Jena, "Parsing-Based Sarcasm Sentiment Recognition in Twitter Data," in ASONAM, 2015. DOI: 10.1145/2808797.2809410.

[24] D. Bamman and N. A. Smith, "Contextualized Sarcasm Detection on Twitter," in Proc. ICWSM, 2015. DOI: 10.1609/icwsm.v9i1.14663.

[25] T. Ptacek, I. Habernal, and J. Hong, "Sarcasm Detection on Czech and English Twitter," in COLING, 2014. DOI: 10.3115/v1/C14-1021.

[26] F. Barbieri, H. Saggion, and F. Ronzano, "Modelling Sarcasm in Twitter, A Novel Approach," in WASSA, 2014. DOI: 10.3115/v1/W14-2611.

[27] Liebrecht, F. Kunneman, and A. van den Bosch, "The Perfect Solution for Detecting Sarcasm in Tweets #not," in WASSA, 2013. DOI: 10.3115/v1/W13-1604.

[28] E. Riloff, A. Qadir, P. Surve, L. De Silva, N. Gilbert, and R. Huang, "Sarcasm as Contrast Between a Positive Sentiment and Negative Situation," in Proc. EMNLP, 2013. DOI: 10.3115/v1/D13-1066.

[29] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient Estimation of Word Representations in Vector Space," arXiv preprint arXiv:1301.3781, 2013. DOI: 10.48550/arXiv.1301.3781.

[30] Ramteke, A. Malu, P. Bhattacharyya, and J. S. Nath, "Detecting Turnarounds in Sentiment Analysis: Thwarting," in ACL Short Papers, 2013. DOI: 10.3115/v1/P13-2150.

[31] Reyes, P. Rosso, and D. Buscaldi, "From Humor Recognition to Irony Detection: The Figurative Language of Social Media," Data & Knowledge Engineering, vol. 74, pp. 1–12, 2012. DOI: 10.1016/j.datak.2012.02.005.

[32] J. D. Campbell and A. N. Katz, "Are There Necessary Conditions for Inducing a Sense of Sarcastic Irony," Discourse Processes, vol. 49, no. 6, pp. 459–480, 2012. DOI: 10.1080/0163853X.2012.688454.

[33] Davidov, O. Tsur, and A. Rappoport, "Semi-Supervised Recognition of Sarcasm in Twitter and Amazon," in Proc. CoNLL, 2010. DOI: 10.5555/1944566.1944584.

[34] O. Tsur, D. Davidov, and A. Rappoport, "ICWSM—A Great Catchy Name: Semi-Supervised Recognition of Sarcastic Sentences in Online Product Reviews," in ICWSM, 2010. DOI: 10.1609/icwsm.v4i1.14031.

[35] P. Rockwell, "Vocal Features of Conversational Sarcasm: A Comparison of Methods," Journal of Psycholinguistic Research, vol. 36, no. 5, pp. 361–369, 2007. DOI: 10.1007/s10936-006-9049-0.

[36] Liu, M. Hu, and J. Cheng, "Opinion Observer: Analyzing and Comparing Opinions on the Web," in WWW, 2005. DOI: 10.1145/1060745.1060797.

[37] S. Attardo, J. Eisterhold, J. Hay, and I. Poggi, "Multimodal Markers of Irony and Sarcasm," Humor, vol. 16, no. 2, pp. 243–260, 2003. DOI: 10.1515/humr.2003.012.

[38] S. L. Ivanko and P. M. Pexman, "Context Incongruity and Irony Processing," Discourse Processes, vol. 35, no. 3, pp. 241–279, 2003. DOI: 10.1207/S15326950DP3503_02.

[39] P. Rockwell, "Empathy and the Expression and Recognition of Sarcasm by Close Relations or Strangers," Perceptual and Motor Skills, vol. 97, no. 1, pp. 251–256, 2003. DOI: 10.2466/pms.2003.97.1.251.