Spectrum of Engineering Sciences
ISSN (e) 3007-3138 (p) 3007-312X
Volume 4, Issue 2, 2026

# USASD: A SENTIMENT AND SARCASM ANALYSIS TECHNIQUE FOR URDU LANGUAGE SCRIPT

**Sidra Hameed[1], Zakia Jalil[*2], Muhammad Nasir[3]**

[1,*2,3]*Faculty of Computing & Information Technology, International Islamic University, Islamabad, Pakistan*

[1]sidra.mscs1138@iiu.edu.pk,[*2]zakia.jalil@iiu.edu.pk, [3]m.nasir@iiu.edu.pk

DOI: https://doi.org/10.5281/zenodo.18830792

**Keywords**
*Urdu Sentiment Analysis, Sarcasm Detection, BERT, CNN-LSTM, T5 Fine-Tuning, XLM-RoBERTa, XGBoost, Deep Learning, Natural Language Processing (NLP).*

**Article History**
*Received: 27 December 2025*
*Accepted: 11 February 2026*
*Published: 28 February 2026*



**Corresponding Author: ***
**Zakia Jalil**

**Abstract**
*Human beings are accustomed to expressing positive or negative comments in their language. The beauty of natural language is in its crispy sarcastic way of expression. Traditionally, sarcasm detection and sentiment analysis are treated as entirely different tasks in Natural Language Processing (NLP), though their relationship is crucial to understanding the contextual meaning. Urdu's richness in morphology and complicated syntax would indeed create something unique in that area. This paper presents a hybrid deep learning model named USASD (Urdu Sentiment Analysis and Sarcasm Detection) incorporating BERT, CNN, LSTM, and fine-tuned T5 for Urdu sentiment analysis and sarcasm detection. It provides contextual embedding, CNN captures the local text pattern, LSTM processes sequential dependence, and T5 puts it in a unified text-to-text paradigm. Pretrained on Urdu datasets fine-tuned using a custom-annotated Urdu tweets dataset, USASD correctly identifies sentiment polarity and nuances of sarcasm with 89% accuracy in sentiment analysis and 83% in sarcasm detection against the XGBoost and XLM-RoBERTa baseline. This work advances NLP for low-resource languages, thus allowing for better text analysis in Urdu for social media monitoring, opinion mining, and multilingual NLP research.*

## INTRODUCTION

The substantially of Urdu has increased during the last decade. The people of subcontinent region use Urdu language for communication, for communication purposes round about 170.2 million people speak the Urdu language over the world (Tahir & Mehmood, 2022), (Qasim et al., 2016), (Hassan et al., 2024),(Ahmad &Edalati, 2022). Urdu is the communication and as well as official language of India's, Azad Kashmir and Pakistan. Mostly explored regions over the Pakistan provides their content in Urdu (Naqvi et al., 2021a). Urdu is the widely used language that has received little attention from the computational linguistics field. Urdu character pattern and shape is categorized from right to left script, it's separation among the alphabet definition is not clear. For example, the expression 'وہمیراگھرہے ' (that is my house) remains clear because in words the spaces are absent. Urdu literature differs in syntactic, orthographic, and morphological features.(Syed et al., 2011)Claims that new and updated techniques for sentiment analysis should be used. Additionally, Urdu has syntax like Hindi but on the other hand has a dissimilar script as well as phonology(Dubinsky et al., 2007).

The rapid growth of social media platforms and digital communication resulted in an exponential rise in user-generated text data, placing sentiment analysis as a core challenge in Natural Language

Processing (NLP) (Naqvi et al., 2021b). Although significant work has been done in NLP for widely spoken languages such as English, the low-resource languages like Urdu are underexplored due to complex morphology, syntactic variations, and contextual dependencies (Ahmad &Edalati, 2022). Sentiment analysis determines the polarity associated with a given text-text, either positive, negative, or neutral-permitting applications that include opinion mining, customer feedback analysis, and sentiment tracking on political issues (Shabbir & Majid, 2024), while sarcasm detection is a very important task that differentiates literal sentiments from sarcastic expressions, which often go against their surface meaning (A et al., 2021). The major problem for accurate sentiment classification arises with sarcasm because it carries implied meaning. Implicitness in meanings leads to the conflict between literal words and the desired sentiment being usually opposite (Amir et al., 2016a). Traditionally, sentiment analysis models are not built to counter sarcasm, leading to misclassifications and unreliable results (Alsaeedi& Khan, 2019).

Detecting sarcasm is especially hard because it is context-dependent, tone-dependent, culturally dependent, and wordplay-dependent, and traditional models for sentiment analysis often fail to classify opinions (Amir et al., 2016a). Transformer-based deep learning models have significantly improved the performance of NLP tasks like sentiment analysis and sarcasm detection, but most research has focused on high-resource languages such as English (Alshamsi et al., 2020). Urdu, like low-resource languages, has been undersampled with an inherent complexity that relates to morphology and syntactical differences as well as a surfeit of mixed-code and casual writing on social media(Bamman & Smith, 2015). Urdu NLP models have heretofore mostly been rooted in rule-based or classical ML techniques and remain poorly suited for semantic depth in implicit sarcasm, leading to a high number of misclassification (Mukhtar & Khan, 2018a).

The purpose of this work is to grow the importance of sarcasm detection and sentiment analysis with Urdu linguistic, the language domain of current study in the context of more advanced communication linguistic processing models (Khan et al., 2021); (Majeed et al., 2020); (Mukhtar & Khan, 2018b); (Naqvi et al., 2021c). By sentiment study helps in discerning the emotional attitude, opinion and feelings in the context, sarcasm recognition represents a further layer of complication. In this study we required trained system to hold natural language shapes and pattern of textual character. The natural language such as Urdu, by its intricate nuances and rich linguistic diversity, represents an exclusive problem for computational models. While concentrating on fine-tuning this system such as T5 for Urdu language, this research objects is to determine the problem in previous works and our contribute is to solve that problem and provide the advancement system for sentiment analysis and sarcasm recognition abilities in Urdu language. However Successful employment of our algorithm can be important inferences for several systems, with social website analyzing, people opinion and customer feedback analysis, determining in Urdu-speaking societies. Moreover, a dynamic nature is the purpose of this research which admits the language evolution and develop the framework which permits continuously adapt the feature of fine-tuned model with emerging language trends in Urdu.

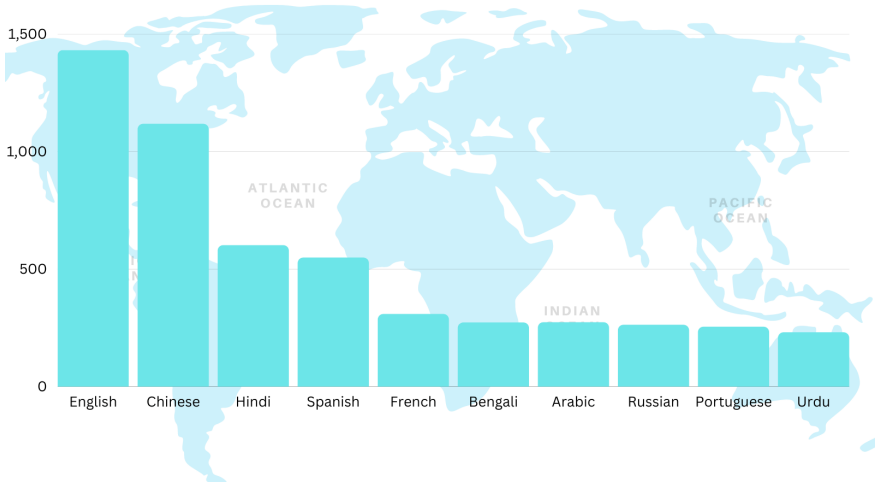


**Figure 1 Most Spoken Languages in the World**

The number of speakers in millions for each of these languages. This means English is the language spoken by maximum numbers, then comes Chinese, Hindi, and then Spanish. Urdu falls among the top ten languages of the world. The fact that Urdu is among the top ten languages primarily spoken indicates that the language is widely spread and used, making it a very valuable language for building AI models, such as sentiment analysis and sarcasm detection. The globalized nature of Urdu-speaking people again confirmed the need to develop stronger NLP tools to advance human computer interaction, especially in terms of text understanding and sarcasm detection abilities for Urdu.

Current existing Urdu sentiment analysis and sarcasm detection systems have issues related to flexibility, efficiency, and accuracy, thus needing robustness in the exact determination of the sentiments. Currently applied techniques lack enough strength in extracting the semantic as well as syntactic structure that results in emotional tone and sarcasm identification from the text, which decreases accuracy in detecting feature extraction (Hassan et al., 2024), (Mukhtar et al., 2018). Many of the classification mechanisms will miss major stylistic inputs and function on raw texts without much representation of contextual features specific to Urdu(Naqvi et al., 2021c). In light of such structural and critical issues, advanced methodologies in feature extraction with higher accuracy in classification outputs, along with increasing reliability and accuracy in sentiment and sarcasm detection for the Urdu language, are urgently needed.

**Natural Language Processing**

Currently AI mostly focus on the understanding of NLP and assessing the languages that humans naturally use to communicate with machines, such as English (Dubinsky et al., 2007), (Chowdhary, 2020). As the number of human machine interactions increases, NLP applications are becoming more popular. These applications can be divided into two categories. Natural Language Processing for Resource-Poor Languages include spellings checkers, machine translations, and also used to grammar checking of natural languages spoken among humans to converse with each other (Anwar et al., 2006). In terms of communicating and manipulating the psychological/conceptual components of natural language, such as text, shape and pattern via NLP is a widely distinct area in the research (Jacquemin, 2001).

**Sentiment Analysis**

Sentiment Analysis (SA) is recognizing how emotions are written in the text and whether the utterances give a negative (undesirable) or positive (desirable) opinion on the topic (Nasukawa& Yi, 2003). Therefore, sentiment

analysis involves identifying emotional expressions, the polarities and strengths of expressions, and their relationships to topics (Nasukawa& Yi, 2003).

**Levels in Sentiment Analysis**

SA is divided into three levels, namely document, sentence, and aspect (Jijkoun et al., 2010). At the document level, the whole text is classified either as positive or negative based on the overall sentiment. When both positive and negative sentences are of equal numbers, then classification is ambiguous. The approach assumes one sentiment in the entire document, which is inappropriate for texts having multiple entities (Jijkoun et al., 2010).

SA, at the sentence level, involves two tasks: Sentiment Classification (SC) and Subjectivity Classification. The Subjectivity Classification task separates the subjective sentences from the objective ones, making it separate the opinions from facts in the actual sentences (Alsaeedi& Khan, 2019). SC classifies the subjective sentences into positive, negative, and neutral (Kharde& Sonawane, 2016).

Aspect-level SA provides a more granular analysis by identifying sentiments associated with specific entities or aspects within a text. This approach allows for a more precise understanding of sentiment across different elements in a document (Schouten &Frasincar, 2016).

**Approaches in Sentiment Analysis**

SA is a vital task in NLP which concerns itself with determining the polarity of sentiment in text (Yadollahi et al., 2017). Early lexicon-based approaches used pre-defined dictionaries of sentiment words for classification but had trouble with context-based nuances such as sarcasm particularly in morphologically complex languages like Urdu (Cambria et al., 2013).

Machine learning-based approaches, including SVM and Naïve Bayes, have improved SA even by using hand-crafted features like word frequency and sentiment score (Dhaoui et al., 2017). Though they could not capture more profound semantic and contextual meanings (Cambria et al., 2013). Deep learning introduced neural networks such as Long Short-Term Memory (LSTM), Recurrent Neural Networks (RNN), and Convolutional Neural Networks (CNN) that deepened contextual understanding by sequential processing(Socher et al., 2013). Recently, transformer models like BERT and mBERT have allowed transfer learning, which greatly strengthens the performance of SA in multilingual settings, especially for low-resource languages.

In this research, BERT, CNN, LSTM, and T5 are used for sentiment and sarcasm classification in Urdu. BERT takes out rich semantic features, CNN captures local patterns, while LSTM is designed for sequential dependencies. These embeddings are leveraged in a unified text-to-text framework by a fine-tuned T5 model for classification (Raffel et al., 2020). Such a hybrid approach can be used to counter challenges that involve the scarcity of annotated data, contextual ambiguities, and linguistic complexities toward enhanced sentiment analysis in the Urdu language.

**Sarcasm Detection**

Sarcasm, a common form of figurative language, often expresses humor or criticism indirectly and poses a considerable challenge for sentiment analysis, especially in social media contexts where emotions and opinions are frequently conveyed non-literally (Hassan et al., 2024) Detecting sentiment analysis, sarcasm accurately is essential for customer feedback processing, and monitoring public opinion, where accurate sentiment interpretation is critical (Joshi et al., 2017), Sarcasm detection is low resource language such as Urdu is particularly complex due to the scarcity of annotated datasets and a unique language structure of Urdu, likes syntax and morphology, which differ significantly from other languages studied in NLP (Hassan et al., 2024) unlike English, Urdu often relies on idiomatic expressions and context-based cues that complicate the task of automatically identifying sarcasm (Bamman & Smith, 2015).

**Cultural Context in Urdu Sarcasm**

Urdu sarcasm is often rooted in cultural references and expressions unique to South Asian society, making it distinct and challenging to detect (Bamman & Smith, 2015). Unlike in English, where sarcasm might use exaggeration or irony, Urdu sarcasm frequently relies on idiomatic expressions, cultural proverbs, and subtle humor that can only be understood with knowledge of social and cultural nuances (Hassan et al., 2024). For example, Urdu speakers may use traditional phrases or polite language with an ironic twist to convey sarcasm, requiring context and cultural familiarity to recognize the intended meaning accurately (Rajadesingan et al., 2015).

Nevertheless, social media, microblogging (such that, maximum 140 text in each tweet) and consists of informal language and also formal essentially join this quite tricky for understanding the user's sentiment and achieve sentiment analysis(Bermingham, 2012). Furthermore, sarcasm has a task in sentiment study and reasons the missed-classification of the opinion of people. Hereafter, in the study sentiment analysis accuracy may be reduced. Society routinely uses sarcasm to mock or convey contempt by the while speaking or sentence. People use positive text in the tweets to expose gloomy moods. In social media irony or sarcasm is very common this time, however, this is huge problem for researchers to detect through sentiment analysis. Furthermore, modern cutting-edge methodologies, data mining techniques and sentiment analysis are pre-disposed to insufficient performances although studying the text on tweets type data (A et al., 2021).

**Approaches in Sarcasm Detection**

NLP used to Sarcasm recognition is the problem able job because this reliance on understanding implied meanings and distinguishing between literal and non-literal expressions. Early approaches used lexicon-based methods, which relied on predefined lists of sarcastic words or rule-based phrases and sentences detection, however, this is very limited in handling context-dependent sarcasm (Davidov et al., 2010b). Additionally, the machine optimization techniques, i.e. Naïve Bayes and Support Vector Machines (SVM), applying handcrafted features like sentiment polarity shifts, punctuation patterns, and part-of-speech tags. However, these methods struggled to effectively capture the nuanced linguistic cues often present in sarcasm (Davidov et al., 2010a). The Approaches of Context-aware to point out the several limitations with incorporating further information, i.e. external context or conversational history, to improve the process of recognition. For effective training the model, we use several techniques and extensive datasets (Amir et al., 2016b), (Wallace et al., 2014). This study uses deep learning algorithm presented by neural network-based techniques, like Recurrent Neural Networks (RNNs), Convolutional Neural Networks (CNNs), and Long Short-Term Memory (LSTM) networks, which allowed for a deeper understanding of sequential relationships and complex patterns in text (Ghosh & Veale, 2016), (Mandal & Mahto, 2019), (Ashok et al., 2020).

These hybrids, namely a combination of transformers along with BERT and T5, with CNN and LSTM are used to realize sarcasm from Urdu text. These use BERT-pretrained embeddings, bringing rich semantic, contextual, CNNs capture locality, and then LSTMs for modeling the sequential dependency present inside the features themselves. Then with fine-tuned T5-classification model enrich the system ability with respect to noticing any nuanced feature available in text or not.

Currently the challenge is to detect the correct features of natural language such as Urdu due to the existing model not being able to resolve the complexity. It is very critical for researchers to fill this gap. There is a lot of work around natural language processing, mainly Urdu language, for researchers except feature detection and classification, such as sentiment analyzing, public opinion, people discussions on social media, in business fields the customer feedback, and communication between persons. The purpose of this work is to resolve these problems by using advanced machine learning algorithms such as sentiment analysis and sarcasm recognition in

Urdu context. To support this inquiry, the following research questions are introduced:

1. How does the implementation of an advanced transformer algorithm **enhance the accuracy** of sentiment analysis and sarcasm detection in Urdu language content?
2. How can the **flexibility and efficacy** of natural language processing models be improved when applied to Urdu text?

This paper presents a hybrid approach combining BERT, CNN, LSTM, and T5 to improve Urdu sentiment analysis and sarcasm detection. BERT captures deep semantic and contextual features that are further processed by CNN and LSTM to identify local patterns and sequential dependencies. T5 fine-tuned for classification further improves the overall accuracy. A custom-labeled Urdu dataset of tweets and phrases is used, providing a valuable resource for future research. Compared with other baseline algorithms, the model in this work improves precision, recall, accuracy, and F1-score remarkably. This paper presents practical values of social media analysis, opinion mining, and Urdu-speaking customers' feedback monitoring, and, accordingly, an open framework of a model adaptable and relevant through language trends updates for Urdu languages.

## Related Work

In this section, the review of the state-of-the-art for the automated classification of sentiment and sarcasm in standard Urdu has been presented. However, in this work for the Urdu language, we had considered academic and research papers maintaining the work on Nastalique, which is a standard script for Urdu.

By going through the literature available online and alongside the relevant academic research portals, we found that very little work done so far had been done regarding the automated classification of sarcasm in South Asian regional languages, such as Bengali, Hindi, Punjabi, etc. In a similar context, based on the available date so far, we found no good work regarding the sarcastic classification available for the text in Urdu. Related work thus reports that there is work on the other languages w.r.t significance.

### Sentiment Analysis

It has been extensively researched in English as well as many other high-resource languages using approaches from the lexicon-based methods to transformer-based models. Low-resource languages researchers have used transfer learning, multilingual embeddings, and even limited labeled datasets. In Urdu, most research has been on applying machine learning techniques, such as SVM and Naïve Bayes, and recently deep learning approaches. These studies are the foundation of what is known and understood about sentiment but often fail to explain complexities such as sarcasm.

(Shabbir & Majid, 2024) discussed deep learning techniques for Urdu sentiment analysis in order to classify public opinion within Urdu-speaking communities. Realizing that there is a rapid increase in Urdu textual data, the authors discussed advanced transformer-based models as an alternative to the traditional machine learning and lexicon-based approach. The authors used the translated Urdu version of the IMDB movie review dataset to compare 1D-CNN, LSTM, and the Multilingual-MiniLM-L12-H384 transformer. It was concluded that the transformer model had shown the highest accuracy of 89.36% and good ability for Urdu sentiment analysis.

(Ali et al., 2024) stressed the need for sentiment analysis in low resource languages like Urdu for understanding public opinion. They ventured out and created a large-scale dataset compiled from various sources (newspapers, articles, social media) and categorized sentiments into positive, negative, neutral, mixed and ambiguous. Their research has found that the RNN model outperforms CNN using the deep learning model, and this has reached 91% accuracy, and thus is useful for Urdu sentiment analysis.

(Naqvi et al., 2021c) addressed the challenges of Urdu sentiment analysis imposed by this morphology-wealth language. They proposed a framework that integrated word-vector representations with deep learning models: LSTM, BiLSTM with Attention (BiLSTM-ATT), CNN, and CNN-LSTM. Their results showed that the BiLSTM-ATT model acquired the best result with 77.9% accuracy and a 72.7% F1-score.

The study also investigates the effect of pre-trained and self-trained embedding models on sentiment classification performance.

(Rehman et al., n.d.)addressed to classifying Urdu-language political texts using machine learning and NLP. This study analyzes over 1,300 tweets by 13 prominent Pakistani political leaders, categorized into six groups, which are associated with policies, campaigns, and opinions. It applies sentiment analysis to understand the political moods, with techniques of data preprocessing such as cleaning, balancing, and stop-word removal. The features are extracted using TF-IDF, and models such as SVM, Decision Tree, XGBoost, and Random Forest are implemented. Besides this, Word2Vec is also applied to perform neural network-based sentiment classification. The study tries to grasp the political sentiments used on social media by leaders.

(Majid et al., 2023) discussed the difficult grammatical nature of the Urdu language as a sentiment analysis challenge. They have put forward a lexicon-based method that relies on a list of certain positive and negative words. The system uses rules of morphological analysis of input text based on lexical categories and prefixes that differentiate sentiments. For a moderate-sized Urdu dataset, they used a rule-based approach to arrive at a very acceptable accuracy rate of 84%.

Their method was tested against existing models and demonstrated improved accuracy, precision, recall, and F1-score, suggesting potential applications in social media monitoring and customer feedback analysis.

**Sarcasm Detection**

Sarcasm detection is a challenging yet essential task in natural language processing, especially for low-resource languages like Urdu, Arabic, Pashto, and Roman-Urdu, which lack substantial exploration in this domain.

(Onan, 2017)pointed out an important challenge in identifying sarcasm in social media text-that of literal versus intended meaning. Their machine learning-based approach is based on lexical and pragmatic features: unigrams, bigrams, and term representations, such as TF-IDF. They experiment and test SVM, Naïve Bayes, and logistic regression classifiers. SVM using the presence of the terms and unigrams achieved a significant accuracy of 89.15%, indicating the feasibility of using SVM as a tool for sarcastic identification.

(S. Khan et al., 2024) pointed the Urdu Sarcastic Tweets (UST) dataset to tackle the challenges of sarcasm detection in low-resource languages. UST and TanzIndicator datasets are used to test different machine learning models, such as SVM, Decision Tree, KNN, and Naïve Bayes. The highest accuracy is obtained by SVM with a value above 78%. The paper suggests that customized methods are required for sarcasm detection and offers the UST dataset as an open-access resource for further development.

(Hassan et al., 2024) present that Sarcasm detection is of immense importance to understanding communication on Twitter, particularly for low-resource languages such as Urdu. Moving away from the rules-based method, this work applies transformer-based models and uses a newly annotated dataset of 12,910 Urdu tweets. A hybrid model of mBERT-BiLSTM-MHA with multilingual BERT combined with BiLSTM and multi-head attention was designed, besides five deep learning models: CNN, LSTM, GRU, BiLSTM, and CNN-LSTM. The hybrid model achieves superior performance with 79.51% accuracy and an 80.04% F1 score, outperforming models using fastText embeddings.

(Farhan et al., 2024) introduce a Bi-LSTM model enriched with pre-trained GloVe word embeddings for the purpose of detecting sarcasm in multi-domain datasets. For the construction of the dataset, three public datasets were merged: Gosh Tweets, News Headlines Dataset, and Sarcasm Corpus v2. GloVe word embeddings extracted contextual and semantic features that were later used by the Bi-LSTM model for detecting sarcasm. The model proposed was able to obtain 86.35% accuracy with a recall, precision, and F1 of 88%. This paper, therefore, is a highlight on the importance of contextual information for the detection of sarcasm while

achieving state-of-the-art performance in various applications to improve the sense of sentiment analysis.

Table 1 highlights the main findings of traditional approaches employed for sentiment analysis and sarcasm detection in Urdu text. Although these conventional models have provided significant insights into the field of sentiment and sarcasm classification, they come with certain inherent limitations, as outlined in Table 1.

**Table 1 Comparative analysis of different techniques for SA and SD**

| Ref. | Objective | Dataset | Method | Result | Limitation |
|---|---|---|---|---|---|
| **(Shabbir & Majid, 2024)** | To develop a framework for sentiment analysis of Urdu text using deep learning models. | IMDB movie review dataset | 1D-CNN, LSTM, and Multilingual-MiniLM-L12-H384 | Transformer model achieved 89.36% accuracy, outperforming other models. | The dataset is limited to movie reviews, which may not generalize to all types of Urdu text. |
| **(Ali et al., 2024)** | To develop a sentiment analysis model for Urdu text using deep learning techniques | Custom dataset collected from newspapers, articles, and social media comments. | CNN and RNN | RNN achieved an accuracy of 91%, outperforming CNN. | Limited to five sentiment labels and may not cover all types of sentiment nuances. |
| **(Naqvi et al., 2021c)** | To propose a framework for Urdu Text Sentiment Analysis (UTSA) using deep learning methods. | Urdu language text data | LSTM, BiLSTM-ATT, CNN, and CNN-LSTM. | BiLSTM-ATT achieved 77.9% accuracy and 72.7% F1 score. | Limited exploration of other advanced deep learning models.<br><br>Reliance on specific embedding models may affect generalizability. |
| **(Rehman et al., n.d.)** | To classify political tweets in Urdu and analyze sentiments using NLP and machine learning models. | 1300+ tweets from 13 famous Pakistani political leaders, categorized into 6 groups. | SVM, Logistic Regression, Decision Tree, XGBoost, Random Forest | Logistic Regression: 85% accuracy SVM: 84% accuracy | Limited to political tweets in Urdu and focused only on specific political figures. |
| **(Majid et al., 2023)** | To propose a morphology-driven approach for Urdu sentiment analysis. | Urdu text dataset | Morphological Rule-based Approach | Accuracy=84% | Limited exploration of other machine learning methods. Relies heavily on predefined lexicon. |
| **(S. Khan et al., 2024)** | Develop tailored sarcasm detection methods for low-resource languages and evaluate ML classifiers for Urdu sarcasm detection. | UST (Urdu Sarcastic Tweets), Tanz-Indicator | SVM, DT, K-NN, LR, RF, NB, and XGBoost | SVM achieved the highest accuracy of 78% | Limited focus on deeper semantic understanding and contextual sarcasm beyond the dataset-specific characteristics. |

| | | | | | |
|---|---|---|---|---|---|
| **(Hassan et al., 2024)** | To develop a hybrid model for sarcasm detection in Urdu tweets by leveraging mBERT and attention mechanisms. | A curated dataset of 12,910 manually annotated Urdu tweets, derived from 19,995 public Urdu tweets. | CNN, LSTM, GRU, BiLSTM, CNN-LSTM. Mbert-BiLSTM-MHA. | mBERT-BiLSTM-MHA achieved 79.51% accuracy and 80.04% F1 score. | Limited to sarcasm in Urdu tweets, which may not generalize to other platforms or multilingual sarcasm. Relies on manually annotated data, which is time-consuming and might introduce human bias. |
| **(Onan, 2017)** | To improve sarcasm detection using lexical and pragmatic features in machine learning. | Twitter messages dataset. | SVM, Naïve Bayes, Logistic Regression, k-NN. | SVM with term-presence and unigrams achieved 89.15% accuracy. | Focused only on specific feature combinations; limited exploration of contextual understanding. |
| **(Farhan et al., 2024)** | To detect sarcastic remarks in multi-domain datasets using contextual information. | Combined dataset from Gosh Tweets, News Headlines, and Sarcasm | Bi-LSTM | accuracy 86.35%, precision 88%, recall 88%, and F1-score 88%. | Limited to sarcasm expressed in text; performance may vary with non-literal or cultural-specific sarcasm |

**Gaps and Limitations**

Even though there have been significant advances on the lines of sentiment analysis and sarcasm detection on high-resource languages like English and even some advancements in related South Asian languages like Hindi, Urdu remains an understudied area on this front. Major problems are limited annotated datasets, low computational power for Urdu NLP, and there being no pre-trained models designed explicitly for the language. Besides, multilingual models fail to focus on the nuances of cultural and linguistic nature, which are of great importance for sentiment and sarcasm. Therefore, this also reflects a critical need for specialized efforts towards Urdu-based sarcasm detection through more advanced approaches, which are to be combined with contextual understanding, linguistic, and cultural insight.

## PROPOSED METHODOLOGY

The proposed methodology for Urdu sentiment analysis and sarcasm detection will be divided into four key components: Datasets, Preprocessing, Feature Extraction, and Fine-Tuning. This framework includes multiple techniques in order to increase the performance of the model for effective handling of Urdu's complexities in language.

Figure 2illustrates the architectural design of our proposed model, which is composed of essential components such as the pre-processing layer, the embedding layer, and the classification layer.

Below, we discuss the motivation of the method we have proposed:

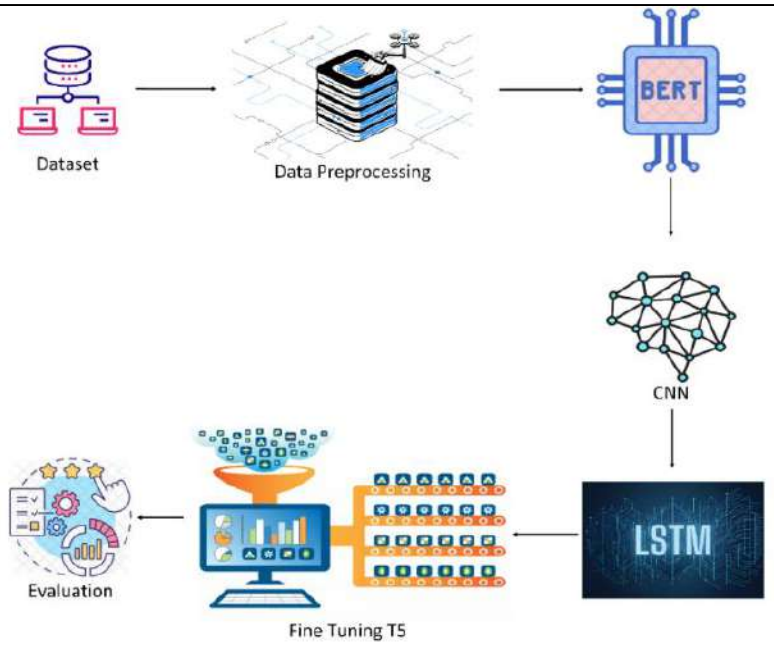


**Figure 2 Proposed Model for USASD**

**Motivation**

This study aims to enhance sentiment analysis and sarcasm detection for the Urdu language by addressing linguistic complexities and advancing computational models. Urdu's intricate nuances pose unique challenges, requiring a system capable of recognizing textual patterns and context. By fine-tuning T5 specifically for Urdu, this research overcomes limitations in previous works, improving sentiment and sarcasm recognition. It has very strong implications in the field of social media analysis, opinion mining, and assessment of customer feedback in Urdu-speaking communities. This work is also toward the employability of NLP systems, allowing them to progress along the changing linguistic trends.

**Dataset**

Choosing the right dataset is essential for efficient sentiment analysis and sarcasm detection. A diversified dataset would give better generalization, reduce bias, and pick up on the linguistic nuances like dialects, idiomatic expressions, and different sentence structures.

This study makes use of open-source datasets for the sake of accessibility and reliability.

We used the Urdu Sarcastic Tweets Dataset, containing 20,004 labeled tweets, which makes it a great resource for training and testing models for sarcasm detection. The dataset allows us to focus on the analysis of sarcasm within the Urdu language, which is a relatively under-explored area in computational linguistics.

For sentiment analysis, we used a very diverse dataset of 7,055 entries derived from different domains-capture, including tweets, news headlines, and weather reports. This diversified dataset allowed our study to comprehensively explore sentiment variation across types of communication, thereby strengthening the strength of our results and enhancing the performance of Urdu sentiment analysis.

**Preprocessing**

Pre-processing is an essential step in NLP projects, as it converts unorganized, raw data into a clean, structured format that allows algorithms to analyze it efficiently.

For our datasets, we will apply the following pre-processing techniques: stop word removal, lemmatization, text conversion to lowercase, elimination of extra white spaces, removal of numbers, and removal of punctuation.

**Cleaning the Dataset**

Both sarcasm detection and sentiment analysis datasets were systematically preprocessed to ensure data integrity and improve machine learning processing.

For sarcasm detection, it retained only columns that are related, which is urdu_text and is_sarcastic, while ensuring to remove any missing values as well. Is_sarcastic column converted into an integer format and standardized urdu_text to Text according to the requirement of the dataset for sentiment analysis. All the above steps gave a structured homogeneous dataset ready to be used in identifying sarcasm.

The text and polarity were retained for sentiment analysis. Removing missing values maintains data consistency and transforms the sentiment labels into a numerical code for further machine learning use: P → 1, N → 0, Neu → 2. Before sending this through the machine learning model, these pre-processed steps were taken to ensure a clean dataset for the model.

**Removal of Stop words**

The most common Urdu stop words include "ایک" (a), "اور" (and), "کا" (of), "میں" (in), "ہے" (is), and "یہ" (this). These words occur frequently in the text but contribute less to the sentiment analysis or sarcasm detection. Removing these stop words resulted in noise reduction and helped the model focus on more meaningful words, thus enhancing pattern recognition and improving the overall performance of the model.

**Punctuation Removal**

The text was further simplified and de-cluttered from the unnecessary marks of punctuation, such as full stops, commas, and exclamation marks. This step further ensured that meaningful words were being analyzed rather than burdening the machine learning model with cluttered and distracted text. With fewer irrelevant elements in the data, the model could analyze linguistic patterns more successfully and yield much better accuracy and performance in tasks like sentiment analysis and sarcasm detection.

**Removal of extra white spaces**

We remove all extra spaces from the text to ensure efficient and clean preprocessing and make it easy for the model to make better predictions.

**Tokenization**

Both sarcasm detection and sentiment analysis datasets were tokenized using the BERT tokenizer. This converted the raw textual data into numerical representations for model processing while preserving the linguistic and contextual nuances. Text was first transformed into string lists to ensure that it is compatible with the tokenizer, which in turn generates tokens using BERT's pre-trained vocabulary. During tokenization, sequences were padded for short texts and truncated for long ones to keep the input dimensions uniform, essential for batch processing and training effectiveness. Also, the tokenizer output comprises input IDs and attention masks, which fit the data to BERT's feature extraction process.

**Embeddings Generation**

BERT-base-multilingual-cased has been applied for embedding generation in sarcasm detection and sentiment analysis tasks. Being a model pre-trained on texts of multiple languages, it is excellent at the processing of various languages, such as Urdu, and capturing relationships between different contextual lines, besides slang and expressions from a culture that are unique. These things make it more ideal for extracting features from Urdu text to obtain relevance in the context of the classification of downstream tasks.

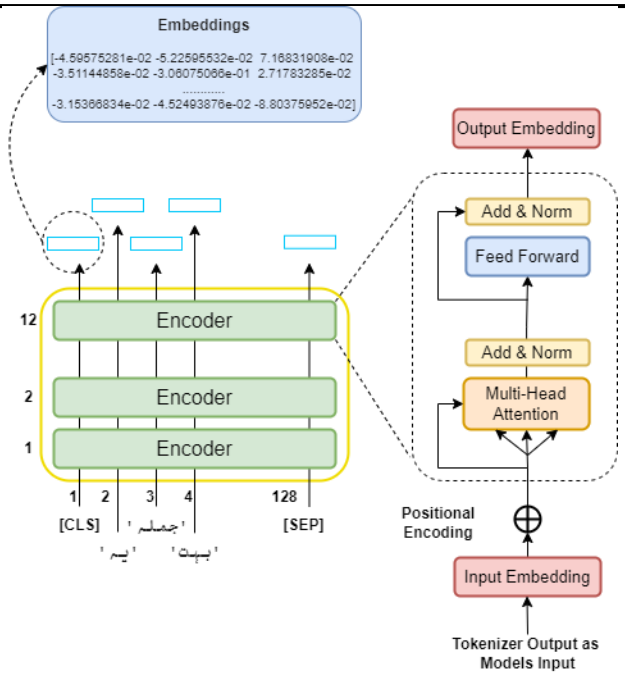


**Figure 3DistilBERT Architecture of embeddings generation in USASD**

Figure 3 describes that the BERT model uses a multi-layer transformer architecture that takes input IDs and attention masks, allowing the output hidden states to yield rich contextual representations. In detail, the summary of the whole input text is represented by the information aggregated at the beginning of the sequence with the [CLS] token, which is the feature vector used for classification in sarcasm detection and sentiment analysis.

During this study, the weights of the model were frozen in inference mode with torch.no_grad() so that feature extraction would not require any further fine-tuning. The extracted features were stored as dense vectors and easily passed to the next stages of classification. The dense vector representations, especially those from the [CLS] token, capture key contextual and semantic information relevant for both sarcasm and sentiment analysis tasks. This unified representation ensures high-quality input for subsequent classification, capturing emotional tone and contextual dependencies across domains like tweets, news, and weather reports.

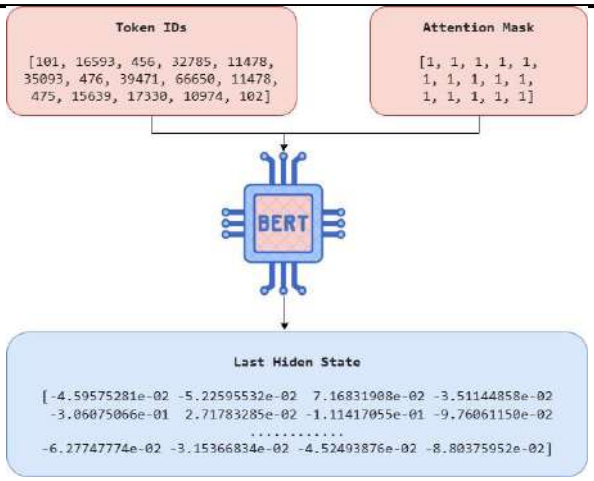


**Figure 4DistilBERT Embeddings for USASD**

Figure 4 shows the generated embeddings using DistilBERT. Each number in the vector represents a specific feature or dimension of the word. These features could capture information such as semantic similarity.

**Feature Extraction Through CNN**

The CNN layer is most important in extracting contextual features that are localized from BERT embeddings. This layer uses several 1D convolutional filters of sizes, for example, 3, 5, and 7, to capture n-grams of various lengths. Its input is the shape of tensor of (batch_size, seq_length, embedding_dim), where every token corresponds to a 768-dimensional vector from BERT. These convolutional filters slide across the sequence, applying their learned weights to generate feature maps that highlight significant local patterns, such as phrases or subword groupings. Following convolution, a ReLU activation function is applied to introduce non-linearity, enabling the model to learn complex features. Outputs of all filters are further concatenated along channel dimensions and create a tensor shape of (batch_size, seq_length, total_filters) by adding all filter outputs together into total_filters.

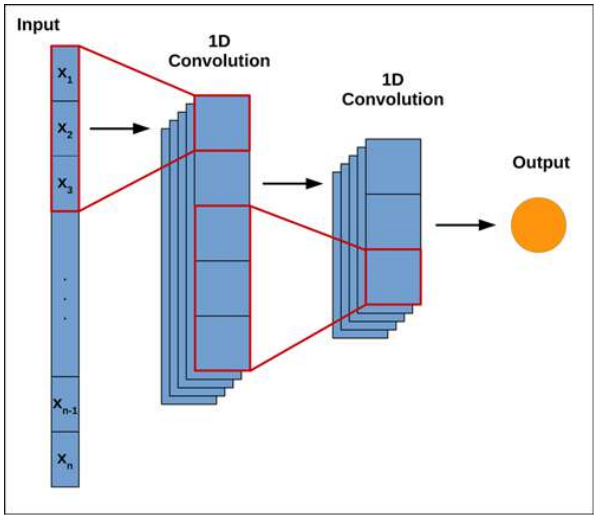


**Figure 5 CNN Architecture**

This architecture captures the local dependencies of the input sequence very effectively. Using filters with different kernel sizes, the CNN layer can learn and represent features at various

granularities. The multi-scale approach enriches the feature representation by capturing a broader range of local patterns and, therefore, improves the ability of the model to understand and process the input text.

**Feature Extraction Through LSTM**

After the CNN layer, BiLSTM is added in the architecture to effectively capture sequential dependencies as well as contextual relationships within the extracted features. This layer utilizes a bidirectional LSTM network that reads the sequence in both forward and backward directions. In this way, the model fully models both past and future contexts for each token within the sequence. The input to this layer is a tensor of shape (batch_size, seq_length, total_filters), where total_filters is the combined output from the previous CNN layer. The LSTM takes this input in sequence and maintains a hidden state that changes dynamically as new tokens are processed, capturing information from previously processed tokens.

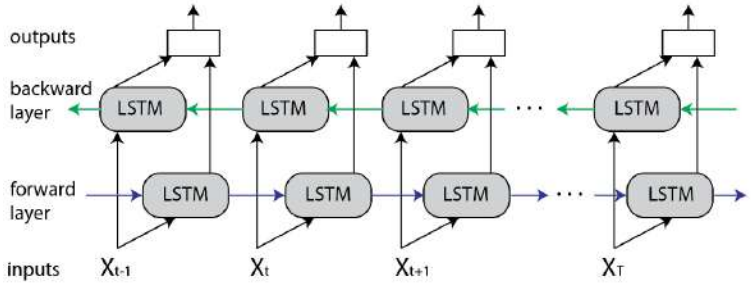


**Figure 6 LSTM Architecture**

The bidirectional nature of the LSTM significantly enhances its ability to capture global dependencies across the entire sequence. The output from the BiLSTM layer is a tensor with dimensions (batch_size, seq_length, 2 × hidden_size), where hidden_size represents the LSTM's hidden state size. Factor 2 arises from the contributions of both the forward and backward passes. To summarize the combined contextual information for the entire sequence, the final hidden state of the BiLSTM is often extracted and utilized for downstream tasks. This layer effectively complements CNN by modeling long-range dependencies and providing rich, sequentially aware representations for subsequent processing and classification tasks.

**Fine Tuning T5**

We considered the T5 model for our experiment: a pre-trained transformer architecture well-suited to doing text-to-text tasks, including sentiment analysis and sarcasm detection. We chose t5-small: this is a relatively lightweight model that was efficient but still handled all our tasks. We found supporting evidence in the model's ability to generalize well on multilingual data as well as our Urdu datasets. As a way to suit T5's text-based input/output paradigm, we used T5Tokenizer to transform the output labels, for example, "positive," "negative," and "neutral," to their text forms to maintain prediction consistency and accuracy within the model framework.

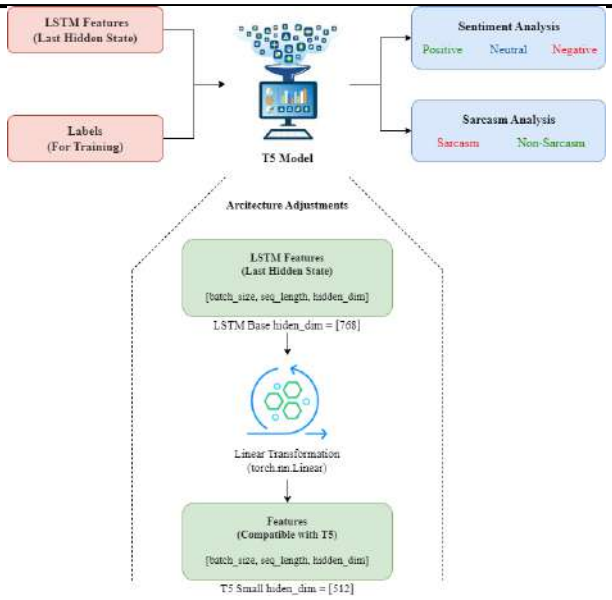


**Figure 7 Architecture Adjustments in USASD**

During fine-tuning, the LSTM-extracted features were incorporated into the T5 model. An important step toward compatibility was dimension alignment between the LSTM-extracted features and the dimensional requirements of the T5 model. Whenever a mismatch existed, a linear transformation was used to adjust the dimensions of the LSTM features so that their semantic and contextual integrity was preserved, as shown in Figure 3.9. This linear layer simply served as an interface between both models, giving T5 proper access to such rich, contextually informed representations from the LSTM without ever posing any dimensional alignment issues. Finally, the T5 model learnt the optimal map for the rich embeddings during joint training with finetuning for the weights in this linear layer.

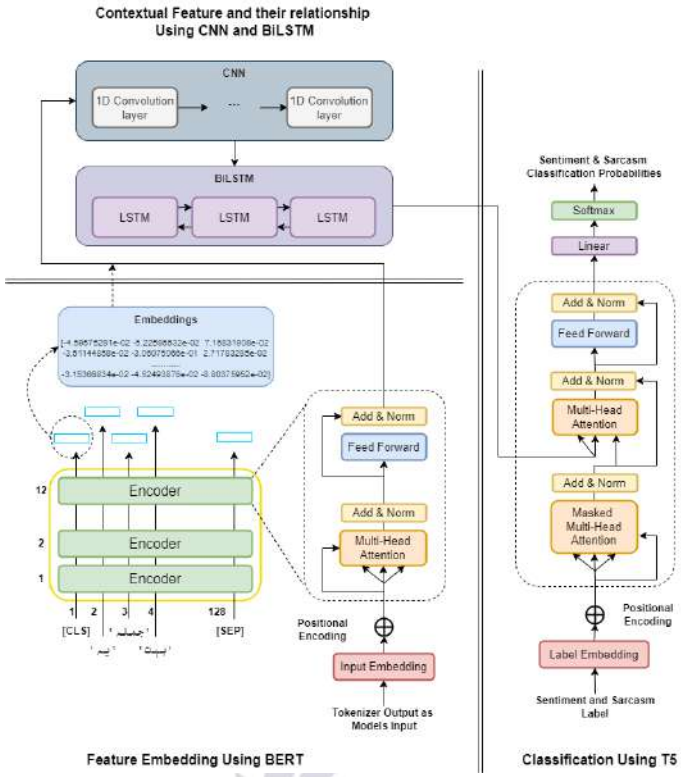


**Figure 8 Complete Architecture: USASD**

The outputs from the BiLSTM are passed into a classification module implemented using T5 transformer, where additional refinement is done over the extracted features with masked attention, multi-head attention, as well as introducing positional encodings to adapt a sequence for classification, followed by the linear layer, softmax function to produce output. It provides an effective combination for local feature extraction, sequential dependency modeling, and powerful transformer-basedcontextualization that produces high-quality predictions on complex text data.

The labels from the dataset were translated to text for compatibility in the text-to-text paradigm presented by T5. For the sentiment-analysis dataset, labels such as "positive," "negative," and "neutral" were replaced with their textual counterparts. Similarly, "sarcastic" and "non-sarcastic" were employed to work with the sarcasm dataset. Further processing of the text used the T5 tokenizer to produce the token IDs that were used as the sequences for target outputs during the training process. This ensured compatibility with T5's architecture and enabled the model to learn to predict the desired textual labels for both sentiment analysis and sarcasm detection tasks.

The AdamW optimizer and a learning rate of $1 \times 10^{-4}$ were utilized to train the model. During the training process, the model computes the cross-entropy loss between the predicted and actual token sequences. The gradients are then calculated in the backward pass via backpropagation, enabling the model to update its weights through iteration. Training was conducted on batches of size 16 to ensure efficient processing.

The T5 model was fine-tuned independently on the sentiment analysis and sarcasm detection datasets. For sentiment analysis, the model learned how to predict textual labels for the sentiment polarity of the given input text. In sarcasm detection, the model learned to classify sarcastic and non-sarcastic text based on contextual clues captured in BERT embeddings.

The loss was monitored at every epoch to track the learning process and make sure that it is optimally training for both tasks.

**Experimental Setup**

The experiments were conducted on a Lenovo ThinkPad equipped with an Intel Core i5-10210U processor (1.60 GHz base clock, 2.10 GHz turbo) and 12 GB of DDR4 RDIMM memory, providing sufficient processing power and memory for computationally demanding tasks. The software environment was based on Python 3.8.5, providing a stable foundation for the machine learning libraries used in the project. This research investigated sentiment analysis and sarcasm detection in Urdu text. Various models, including T5, XGBoost, XLM-RoBERTa, and a hybrid approach combining BERT, CNN, LSTM, and T5, were explored. The hybrid approach demonstrated superior performance, effectively leveraging the strengths of both traditional neural network architectures (CNN, LSTM) and modern transformer models (BERT, T5) to capture intricate semantic and contextual nuances within the Urdu text. This synergistic combination resulted in enhanced accuracy for both sentiment analysis and sarcasm detection tasks.

**Train/Test split**

For our experiment, we split the datasets into two sections: a training set and a testing set. Specifically, we allocated 80% of the data for training and 20% for testing, utilizing the train_test_split function from Python's scikit-learn library.

**Training**

In our method, 80% of the data was assigned to training and would provide sufficient examples for the model to learn from. The model was repeatedly trained on this dataset to refine its understanding and extract hidden features and patterns. This iterative process allowed the model to gradually learn the relationships between input features and target labels. To make sure it was robust and generalized, the training set had diverse examples for all different contexts in the data set. That way, this diversity allowed the model to extract more complex patterns and features than otherwise would be possible and enabled the best generalization seen in sentiment analysis and sarcasm detection tasks on unseen data.

**Cross-validation of models**

Cross-validation was used to obtain an unbiased evaluation of the model's performance by systematically training and testing it on different subsets of the data. This technique divided the dataset into multiple folds and iteratively trained and validated the model on different combinations of these folds. At each fold, training and validation metrics have been recorded as a historical object, which helps to visualize performance over epochs with potential overfitting or underfitting problems. This makes the approach overall very comprehensive-through cross-validation combining detailed logging with visualization-provided valuable insights regarding the model performance, allowing better adjustments in terms of the train process and related hyperparameters.

**Testing**

As mentioned earlier, we divided our dataset into training and testing sets. The testing set is utilized to evaluate the model's performance by determining how well it generalizes to new, unseen data. The outcomes from this evaluation are recorded for subsequent visualization and analysis.

**Evaluation of models**

At this stage, we assess how our deep learning models perform by analyzing the test results using statistical methods. The following metrics are used for evaluation purposes: accuracy, precision, recall and f1-score.

**Visualization of results**

The results of each model are represented using a graphical representation. Matplotlib is used in Python for this purpose. We use Matplotlib to visualize the results generated by models.

**Sentiment Analysis using T5**

As a starting point, the pre-trained "t5-small" model was utilized, and from there, transfer learning was allowed to adapt to the specific nuances of the Urdu language. The input text data was cleaned to remove any punctuation, extra spaces, and leading/trailing spaces to maintain the quality and consistency of the data. The dataset was divided into training and testing sets. A custom PyTorch Dataset class was designed to handle the data efficiently for preprocessing with the T5 model. The tokenizer attached to the T5 model was used to turn the Urdu text into numerical input representations, with padding and truncation to a maximum sequence length of 128. The training was performed by using the Trainer class from the Hugging Face Transformers library. Key training parameters were set at a batch size of 16, 10 epochs, and a warmup period of 1000 steps. The Trainer used AdamW optimizer in an implicit form. Weight decay of 0.01 was also applied. Evaluation was conducted at the end of each epoch, and based on the accuracy of evaluation, the best model was saved. The compute_metrics function calculated accuracy, precision, recall, and F1-score so that the whole performance of the model could be evaluated. The whole process of training was checked and metrics were noted, which can be used to analyze how much the model was learning Lastly, the trained model was evaluated against the test set using the evaluate_model function and the same set of metrics was calculated and reported This method can assess the strength of fine-tuning T5 for the task of Urdu sentiment analysis.

**Sarcasm Detection using T5**

It took the pre-trained "t5-small" model as the starting point, applying transfer learning to adapt it to the specific flavor of Urdu. The input data was sourced from a CSV file and cleaned to remove any punctuation, extra spaces, and leading/trailing spaces. Rows with missing values were dropped. The 'is_sarcastic' column was converted to integers, and the Urdu text column was converted to string type. Then, a random split was used to divide the dataset into training and testing sets with a test size of 20% and a fixed random state of 42 for reproducibility. A custom PyTorch Dataset class was created, named SarcasmDataset, which managed the data and processed it by converting Urdu text into numerical representations with the T5 tokenizer, with padding and truncation to a maximum sequence length of 128. The T5 model was fine-tuned on the Trainer class from the Hugging Face Transformers library to classify the samples as sarcastic or not sarcastic. Training configuration is set as follows: epochs of 10, batch size is 16 both for training and evaluation, the warmup step is 1000, weight decay of 0.01. Implicit use of AdamW optimizer in the Trainer. Evaluate at the end of each epoch and save the model with the best evaluation accuracy. The compute_metrics function calculated accuracy, precision, recall, and F1-score (using the 'binary' average for the two classes), providing a comprehensive evaluation of the model's performance. The training progress was monitored and logged. Finally, the trained model was evaluated on the held-out test set using the evaluate_model function. The model was moved to the available device (GPU if available, otherwise CPU) before evaluation. The same metrics (accuracy, precision, recall, and F1-score) were calculated and reported for the test set as well. In this way, the effectiveness of fine-tuning T5 was evaluated for Urdu sarcasm detection.

**Sentiment Analysis using XGBoost**

For the XGBoost model, TF-IDF vectorization was used to convert the Urdu text data into numerical features. A TfidfVectorizer with a maximum of 1000 features were used to capture the importance of words within the corpus. Then, the obtained TF-IDF vectors were utilized to train an XGBoost classifier. Data was converted into the DMatrix format, which is the optimized input format for XGBoost. This was done using the xgboost.train function. The objective function was set as "multi:softmax" since it was a multi-class classification problem, i.e., it was positive, negative, and neutral. Then, the number of classes is set to 3. In the training procedure, multi-class log loss was used as well as

multi-class error rate (merror) as the metrics. The number of boosting rounds was set at 100. The training process included evaluation on both the training and test sets, with the results stored in the evals_result dictionary. This allowed monitoring of the model's performance during training and visualization of both loss and accuracy across boosting rounds. After training, predictions on the test set were done, and standard classification metrics - accuracy, precision, recall, and F1-score, using a 'weighted' average- reported to measure the performance of the model.

**Sarcasm Detection using XGBoost**

For sarcasm detection, an XGBoost classifier was trained using TF-IDF vectorized Urdu text. A TfidfVectorizer with a maximum of 1000 features were used. The resulting TF-IDF vectors were used as input features for the XGBoost model. The data was converted to the DMatrix format for optimized XGBoost input. The objective of the model was set to "binary:logistic" for the binary classification task (sarcastic or not). The metrics used in training were log loss (logloss) and classification error (error). Training was done over 100 boosting rounds. Training evaluated on the training and test sets and kept track of results, so the loss and accuracy can be monitored and visualized across boosting rounds. Predictions on the test set were made first by obtaining probability scores, which are then converted into actual binary predictions using a threshold of 0.5. Standard metrics for binary classification (accuracy, precision, recall, and F1-score with the 'binary' average) were used to evaluate model performance.

**Sentiment Analysis using XLM RoBERTa**

For sentiment analysis using XLM-RoBERTa model, the xlm-roberta-large pre-trained model was utilized for fine-tuning. Input text was tokenized using AutoTokenizer associated with the XLM-RoBERTa model. Padding and truncation were used to limit it to a sequence length of 128. Custom SentimentDataset class was applied to manage tokenized data as well as their corresponding labels. The data was loaded in PyTorchDataLoader objects that handle batch loading efficiently during training and evaluation phases. A batch size of 16 is used while the model was fine-tuned using the AdamW optimizer set at a learning rate of 3e-5 with weight decay set to 1e-2, and a total of 8 epochs through applying a linear scheduler with a warmup period of 100 steps. Iterate through the training data in batches; calculate the loss using the CrossEntropyLoss criterion and do backpropagation by updating the model's weights using the optimizer and scheduler. The training accuracy was calculated and logged for each epoch. GPU memory was explicitly cleared after each epoch using torch.cuda.empty_cache(). Finally, the trained model was evaluated on the test set. The evaluation process consists of iterating through test data for predictions and calculating standard classification metrics (accuracy, precision, recall, and F1-score using a 'weighted' average) in order to evaluate the model.

**Sarcasm Detection using XLM RoBERTa**

For the detection of sarcasm, a fine-tuned XLM-RoBERTa model was used. The xlm-roberta-large pre-trained model was used. Tokenization of the text was done using AutoTokenizer with padding and truncation to a maximum length of 128. A SarcasmDataset class was made custom to handle the data and labels after being tokenized. Data loaders were defined for efficient batching during training and evaluation with a batch size of 16. The model is adapted for binary classification by altering the classifier head to output just one logit. The loss function used here is BCEWithLogitsLoss, which wraps a sigmoid layer and the binary cross-entropy loss for improved numerical stability. The AdamW optimizer was applied with a learning rate of 3e-5 and weight decay of 1e-2. A linear scheduler with a 100-step warmup was applied to the learning rate adjustment; 8 epochs of training were performed. The training includes iterations over the training data, computation of the loss, backpropagation, and update of the weights. Training accuracy was also calculated and logged per epoch. The GPU memory was cleared after each epoch using

torch.cuda.empty_cache(). During evaluation on the test set, predictions were made by passing the input through the model, applying a sigmoid function to the single logit output, and then thresholding at 0.5 to obtain binary predictions. Standard binary classification metrics (accuracy, precision, recall, and F1-score) were then computed.

**Sentiment Analysis using USASD Model**

This multi-class sentiment analysis model, using positive, negative, and neutral, is built by combining the BERT, CNN, LSTM, and T5 components. The contextualized word embeddings are generated using the bert-base-multilingual-cased model. Then, the processed embeddings pass through a CNN layer with filter sizes 3, 5, and 7 to extract local n-gram features. Then, the output of the CNN is passed through a bidirectional LSTM to capture long-range dependencies. The final hidden state of the LSTM is fed into a pre-trained T5ForConditionalGeneration model. Then, the output of the T5 encoder is fed into a linear classifier for predicting the sentiment label, such as positive, negative, or neutral. It was trained using the AdamW optimizer with 3e-5 learning rate and weight decay of 1e-2. A linear scheduler with 100-step warmup and 8 epochs were used. The CrossEntropyLoss function was utilized. Standard training procedures included loss, backpropagation, and update weights steps. Training accuracy was measured at each epoch. The trained model was assessed on the test set. Standard classification metrics (accuracy, precision, recall, and weighted F1-score) were provided. This architecture was supposed to leverage both BERT, CNNs, LSTMs, and T5 to supervise multi-class sentiment classification. It uses BERT for precontextualization, CNNs for local features, LSTMs for sequential information, and T5 for further contextual refinement before final classification as having a positive, negative, or neutral sentiment category.

**Sarcasm Detection using USASD Model**

This model uses the architecture of a CNN, LSTM and T5 for a sarcasm detector on top of a pre-trained BERT model, with bert-base-multilingual-cased, to provide contextualized word embeddings. A CNN layer with 3, 5, and 7 size filters processes those to capture the local n-gram features; and the CNN output is further fed into the bidirectional LSTM to learn the long-range dependencies. The last hidden state of the LSTM is projected to the pre-trained T5 model. Then, the final hidden state of the T5 encoder is used for classification. Finally, a linear classifier is applied to the output of the T5 encoder to predict the sarcasm label. The model was trained using the AdamW optimizer with a learning rate of 3e-5 and a weight decay of 1e-2. It had a linear learning rate scheduler, 100-step warmup, and 8 training epochs. The loss function was CrossEntropyLoss. During training, the process involved computation of loss, backpropagation, and weights update for the model. Accuracy on training set was monitored and logged at every epoch. It was then evaluated on the test set, along with standard classification metrics: accuracy, precision, recall, and F1-score with 'weighted' average. This architecture looks to combine BERT, CNNs, LSTMs, and T5 strengths for better sarcasm detection. It aims to leverage the BERT as an initial contextualizer, use CNNs to extract local features, apply LSTMs on sequential information, and finally leverage T5 in further contextual refinement before final classification.

**Training Parameters**

Table 2 summarizes the hyperparameter settings used for each of the models tested in this paper. For the deep learning models, namely T5, XLM-RoBERTa, and the hybrid BERT+CNN+LSTM+T5, a consistent set of core hyperparameters is used wherever appropriate. All of them made use of the AdamW optimizer with a fixed learning rate of 3e-5 as well as a weight decay of 0.01. A linear learning rate scheduler with a warmup period of 100 steps was used to optimize training. The maximum sequence length for tokenization was set at 128, while batch size during training and evaluation was fixed at 16. For all three models, epochs were set at 10 for T5 and at 8 for XLM-RoBERTa and the hybrid model. The hybrid model also utilized

architectural hyperparameters, such as filter sizes of 3, 5, and 7 for the CNN layer and a hidden size of 256 for the LSTM layer.

**Table 2 Training Parameters**

| Model | Epochs | Batch Size | Learning Rate | Optimizer | Weight Decay | Warmup Steps |
|---|---|---|---|---|---|---|
| **T5** | 10 | 16 | 3e-05 | AdamW | 0.01 | 1000 |
| **XGBoost** | 100 | N/A | N/A | Default (XGBoost) | N/A | N/A |
| **XLM-RoBERTa** | 8 | 16 | 3e-05 | AdamW | 0.01 | 100 |
| **BERT+CNN+LSTM+T5** | 8 | 16 | 3e-05 | AdamW | 0.01 | 100 |

For the XGBoost model, which is fundamentally different from the deep learning models, a separate set of parameters was used. Training for XGBoost does not apply epochs or batch sizes in the same way, and the learning rate is handled implicitly within the boosting process. The main parameter for XGBoost was num_boost_round, which was set to 100, effectively controlling the number of boosting iterations. Evaluation metrics used for XGBoost were multi-class log loss (mlogloss) and multi-class error rate (merror). These parameter settings were guided by standard practice in the individual model domains and, in some cases, preliminary experimentation. A more extensive search of the hyperparameter space might have led to further performance gains, but the selected configurations proved a good foundation for comparing the relative merits of the various model architectures for the Urdu sentiment analysis task.

## Results and Discussions

This section outlines performance for each of the models in this study by reporting results found for Urdu sentiment analysis. For each approach, we list key metrics consisting of accuracy, precision, recall, and F1-score with a view of providing an expansive analysis of relative strengths and weaknesses of this task. These results demonstrate the relative efficacies of architecture and pretraining strategies in this low-resource scenario for sentiment classification.

### Training and Testing Results of Sentiment Analysis using T5

This section presents the training and testing outcomes of T5 model applied to Urdu sentiment analysis dataset.

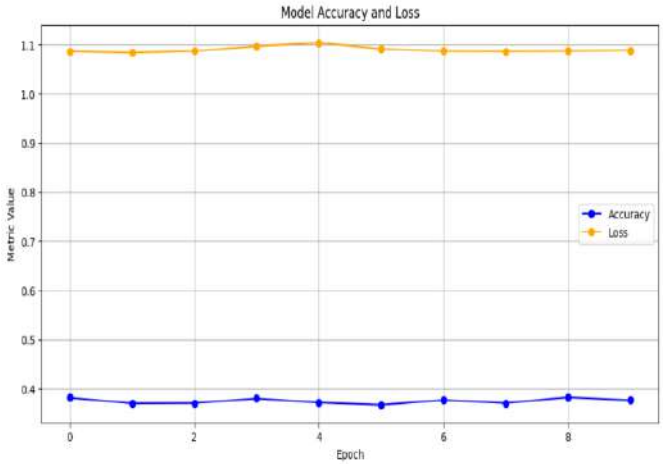


**Figure 9 SA T5 Accuracy and Loss**

The training performance, as reflected in the plotted metrics in figure 9, shows that the loss of the model was persistently high at about 1.1 for all epochs, and the accuracy did not improve much beyond 0.4. This means that the model was not able to effectively minimize the loss and learn meaningful patterns from the dataset. The fact that accuracy does not improve much indicates possible problems like underfitting, lack of training data, or suboptimal hyperparameter tuning.

**Table 3 SA T5 Evaluation Metrics**

| Metrics | Value |
|---|---|
| **Accuracy** | **0.3830** |
| **Precision** | 0.3774 |
| **Recall** | 0.3830 |
| **F1-Score** | 0.3715 |

The test dataset indicates a moderate performance of the model with an accuracy of 38.30%, precision of 37.74%, recall of 38.30%, and a margin of an F1-score of 37.15, as shown in table 3. Therefore, these values indicate that the model has the efficiency to provide some correct predictions but cannot get higher overall performance due to the complexity of sentiment analysis and sarcasm detection in the Urdu text. The relatively low precision and F1-score indicate that the model is not distinguishing between the sentiment classes effectively, possibly because of overlapping features or class imbalance in the dataset.

**Training and Testing Results of Sarcasm Detection using T5**

This section presents the training and testing outcomes of T5 model applied to Urdu sarcasm detection dataset.

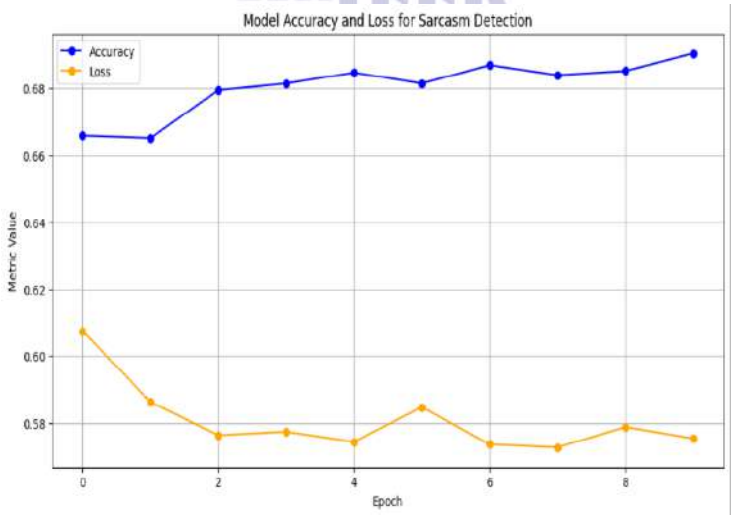


**Figure 10 SD T5 Accuracy and Loss**

The graph of the T5-based model accuracy and loss trends in training for the task of detecting sarcasm in Urdu text is presented in figure 10. The accuracy, depicted as a blue line, improves uniformly over epochs with an increase starting at about 0.66 up to around 0.69 at the last epoch. Such an upward trend reflects that the model learns over time. On the other hand, the orange line shows the loss, which drops drastically in the early epochs, from approximately 0.60 to below 0.58, indicating effective optimization during the early stages of training. The convergence of these metrics indicates that the model was fine-tuned

efficiently with improved stability in performance during later epochs.

**Table 4 SD T5 Evaluation Metrics**

| Metrics | Value |
|---|---|
| **Accuracy** | **0.6906** |
| **Precision** | 0.6496 |
| **Recall** | 0.7892 |
| **F1-Score** | 0.7126 |

The evaluation metrics of the proposed sarcasm detection model is shown in table 4. The accuracy of this model is found to be 0.6906. A precision of 0.6496 means that there is significant reduction of false positives by the model, where detected sarcasm was relevant. The recall of 0.7892 reflects the model's ability to capture most of the sarcastic instances, and the F1-score of 0.7126 balances precision and recall, giving a comprehensive measure of the model's performance.

**Training and Testing Results of Sentiment Analysis using XGBoost**

This section presents the training and testing outcomes of XGBoost model applied to Urdu sentiment analysis dataset.

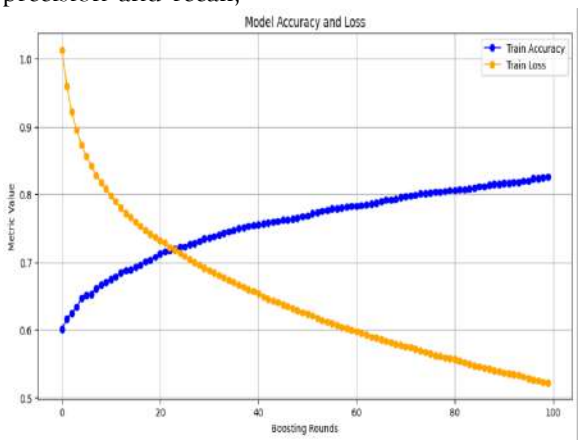


**Figure 11 SA XGBoost Accuracy and Loss**

Figure 11 plots display the XGBoost model's training accuracy and loss patterns at every boosting round over 100 rounds. While training is progressing, the blue curve for accuracy steadily rises to indicate an increased ability to predict the right sentiment labels for this model. In addition, training loss, given in the orange curve, presents an immediate and rapid drop within the first boosting rounds and decreases smoothly thereafter to reveal that the model is properly reducing the error on training. The convergence behavior in the plot indicates that the model is stable as it learns the underlying patterns in the data, balancing accuracy and loss over the rounds of boosting.

**Table 5 SA XGBoost Evaluation Metrics**

| Metrics | Value |
|---|---|
| **Accuracy** | **0.6660** |
| **Precision** | 0.6966 |
| **Recall** | 0.6660 |
| **F1-Score** | 0.6676 |

Table 5 indicates a balanced classification outcome for the model. About two-thirds of the instances were correctly predicted with an accuracy of 66.60%. With the precision score of 69.66%, it shows that the positive predictions made by the model are relatively good, though there is scope for further reduction of false positives. Recall is at 66.60%, implying that the model recognizes approximately two-thirds of the actual positive instances, and some actual positives may have been overlooked. F1-score is 66.76%, indicating a good trade-off between precision and recall, with scope for improvement that will further enhance the overall performance.

**Training and Testing Results of Sarcasm Detection using XGBoost**

This section presents the training and testing outcomes of XGBoost model applied to Urdu sarcasm detection dataset.

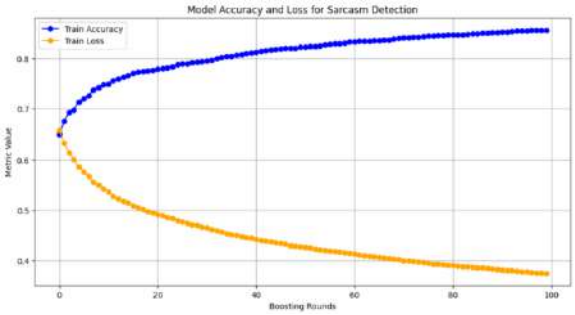


**Figure 12 SD XGBoost Accuracy and Loss**

The graph in figure 12displays the training accuracy and loss curves of the model over 100 rounds of boosting in the case of sarcasm detection. The blue curve displays training accuracy, which has a steady upward trend with each round of boosting, eventually becoming stable at about 80%. This shows the model's capability to predict correctly in the training set. The orange curve shows the training loss, which decreases steadily during the boosting process and stabilizes at about 0.4. Loss reduction shows that the model learns efficiently and converges to an optimal solution. Together, these trends indicate a well-trained model with effective learning during the boosting process.

**Table 6 SD XGBoost Evaluation Metrics**

| Metrics | Value |
|---|---|
| **Accuracy** | **0.7970** |
| **Precision** | 0.8041 |
| **Recall** | 0.7701 |
| **F1-Score** | 0.7867 |

This means that the model was able to classify a great deal of the dataset with a high accuracy of 79.70% as shown in Table 6. A precision of 80.41% shows high reliability in the positive predictions as there are fewer false positives. A recall of 77.01% implies that the model can capture the most actual sarcastic instances, though it missed some true positives. With an F1-score of 78.67%, the precision-recall balance gives the impression that the model, overall, does its job well in performing sarcasm detection.

**Training and Testing Results of Sentiment Analysis using XLM RoBERTa**

This section presents the training and testing outcomes of XLM RoBERTa model applied to Urdu sentiment analysis dataset.

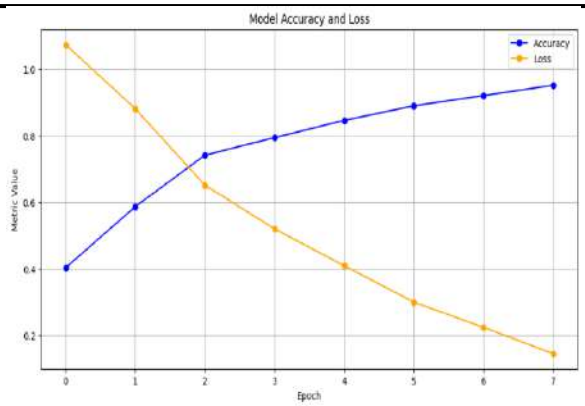


**Figure 13 SA XLM-RoBERTa Accuracy and Loss**

Figure 13 shows the training performance of the fine-tuned XLM-RoBERTa model on sentiment analysis for accuracy and loss over eight epochs. The trend in accuracy steadily increased with the epochs, demonstrating that the model was able to better classify the sentiments with the continued training. Simultaneously, the loss decreased significantly, demonstrating convergence of the model and improved optimization of the classification task. These results validate the effectiveness of the training process since the model is able to achieve balanced learning and consistent performance improvements during the training phase. The visual representation shows the progression toward achieving optimal accuracy while minimizing the error rate.

**Table 7 SA XLM-RoBERTa Evaluation Metrics**

| Metrics | Value |
|---|---|
| **Accuracy** | **0.8113** |
| **Precision** | 0.8130 |
| **Recall** | 0.8113 |
| **F1-Score** | 0.8118 |

Table 7 indicates that the fine-tuned XLM-RoBERTa model gave good performance for sentiment analysis at 81.13%. This was augmented by precision, which came in at 81.30%, and indicates the strength of the model to accurately classify positive instances. At 81.13%, recall depicts the capacity of the model to identify instances correctly, whereas an F1-score of 81.18% is indicative of harmonic balance between precision and recall.

**Training and Testing Results of Sarcasm Detection using XLM RoBERTa**

This section presents the training and testing outcomes of XLM RoBERTa model applied to Urdu sarcasm detection dataset.

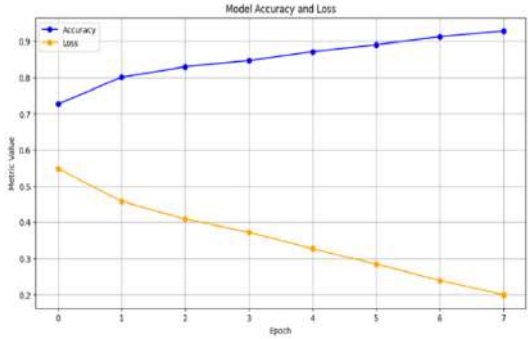


**Figure 14 SD XLM-RoBERTa Accuracy and Loss**

The graph of the training process of the sarcasm detection model clearly showed in figure 14 that it kept improving its accuracy and steadily decreased the loss from epoch 8. Accuracy in the blue line has been shown to increase stepwise and went beyond 90% at the final epoch. Meanwhile, the orange line describes the loss which has decreased progressively from around 0.6 to below 0.2, which reflects that the model learned effectively. These trends reflect the robustness of the training process and suggest that the model optimized its parameters appropriately to enhance its performance on the task of detecting sarcasm in Urdu text.

**Table 8 SD XLM-RoBERTa Evaluation Metrics**

| Metrics | Value |
|---|---|
| **Accuracy** | **0.8059** |
| **Precision** | 0.6895 |
| **Recall** | 0.8031 |
| **F1-Score** | 0.7425 |

The evaluation metrics of the model suggest that the sarcastic detection model is very efficient on the test dataset. Here, it's observed that the accuracy of the model is 80.59%. This implies its overall correctness of classifying a text as either sarcastic or non-sarcastic. A precision value of 68.95% shows the reduction of false positive, while recall value of 80.31% demonstrates most of the sarcastic samples. F1-score is at 74.25%, and it indicates reliable performance when handling the trade-off between precision and recall.

**Training and Testing Results of Sentiment Analysis using USASD Model**

This section presents the training and testing outcomes of USASD model applied to Urdu sentiment analysis dataset.

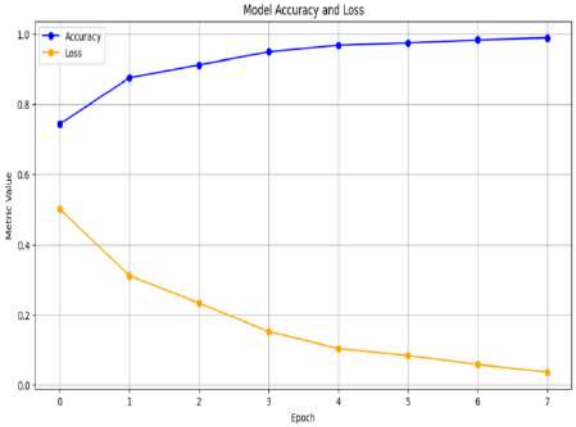


**Figure 15 SA USASD Model Accuracy and Loss**

Figure 15 shows the performance metrics of the USASD model across 8 training epochs, which shows the progress of accuracy and loss across the epochs. Accuracy (blue curve) increases constantly with every epoch and reaches near-optimal values during training. Loss (orange curve) decreases steadily, indicating proper convergence of the model. These trends indicate that the model learns the task appropriately, maintaining a good balance between minimizing loss and maximizing classification accuracy. This performance validates the robustness of the hybrid architecture, and the preprocessing strategies employed.

**Table 9 SA USASD Model Evaluation Metrics**

| Metrics | Value |
|---|---|
| **Accuracy** | **0.8956** |
| **Precision** | 0.8965 |
| **Recall** | 0.8956 |
| **F1-Score** | 0.8958 |

The evaluation metrics of the proposed hybrid model shown in table 9 further confirm its efficacy in sentiment and sarcasm classification tasks. With an accuracy of 0.8956, the model shows a very good ability to classify instances accurately. The precision score of 0.8965 reflects the model's capability to make very precise positive predictions, while the recall of 0.8956 indicates that it is very reliable in terms of identifying all relevant instances. Finally, the F1-score of 0.8958 indicates a good balance between precision and recall, demonstrating the robustness of the model and its applicability in real-world scenarios for dealing with complex linguistic nuances such as sentiment and sarcasm detection.

**Training and Testing Results of Sarcasm Detection using USASD Model**

This section presents the training and testing outcomes of USASD model applied to Urdu sarcasm detection dataset.

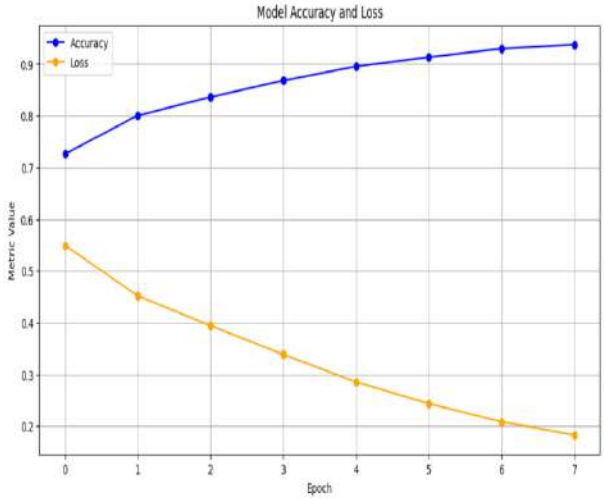


**Figure 16 SD USASD Model Accuracy and Loss**

Figure 16 of the training accuracy and loss for 8 epochs shows a steadily improving accuracy line, peaking at about 0.9 at the last epoch. The loss continuously decreases and hence demonstrates the proper error minimization by the model. Thus, the above two lines show the model is indeed converging well and learning properly.

**Table 10 SD USASD Model Evaluation Metrics**

| Metrics | Value |
|---|---|
| **Accuracy** | **0.8304** |
| **Precision** | 0.8374 |
| **Recall** | 0.8304 |
| **F1-Score** | 0.8299 |

Table 10 shows that the accuracy of the model on the test set was 0.8304, which means that the model made the correct prediction for about 83% of the predictions. A precision score of 0.8374 implies that the model could make positive predictions with high confidence. The recall score is 0.8304, indicating the correct retrieval of instances by the model. So, overall effectiveness in sarcasm detection on Urdu text confirms the effectiveness of the proposed architecture.

**Classification Report Analysis**

This section discusses a complete classification report in terms of performance evaluation of several machine learning and deep learning techniques using the Urdu language dataset on sentiment analysis as well as sarcasm detection. Specifically, we consider T5, XGBoost, XLM-RoBERTa, and a hybrid architecture BERT+CNN+LSTM+T5. We then analyze the results obtained for each individual model and provide a comparative overview across all techniques, yet related, natural language processing challenges.

**Sentiment Analysis Comparison**

Results prove that the USASD outperformed all the other methods with respect to all the above-mentioned measures of evaluation - accuracy, precision, recall, and F1-score - highlighting its better competency in dealing with the intricacies of Urdu sentiment. The hybrid model performs better because of the ability of the CNN and LSTM layers to accommodate sequential dependencies and local patterns along with the feature extraction capacity of BERT for rich contextual understanding and fine-tuning T5 to emphasize task-specific results. Such results highlight the need for deep learning and transformer-based approaches to identify suitable solutions for complex NLP tasks in low-resource languages like Urdu.

**Table 11 Sentiment Analysis Model Comparison**

| | Accuracy | Precision | Recall | F1-Score |
|---|---|---|---|---|
| **T5** | 0.3830 | 0.3774 | 0.3830 | 0.3715 |
| **XGBoost** | 0.6660 | 0.6966 | 0.6660 | 0.6676 |
| **XLM RoBERTa** | 0.8113 | 0.8130 | 0.8113 | 0.8118 |
| **USASD** | **0.8956** | **0.8965** | **0.8956** | **0.8958** |

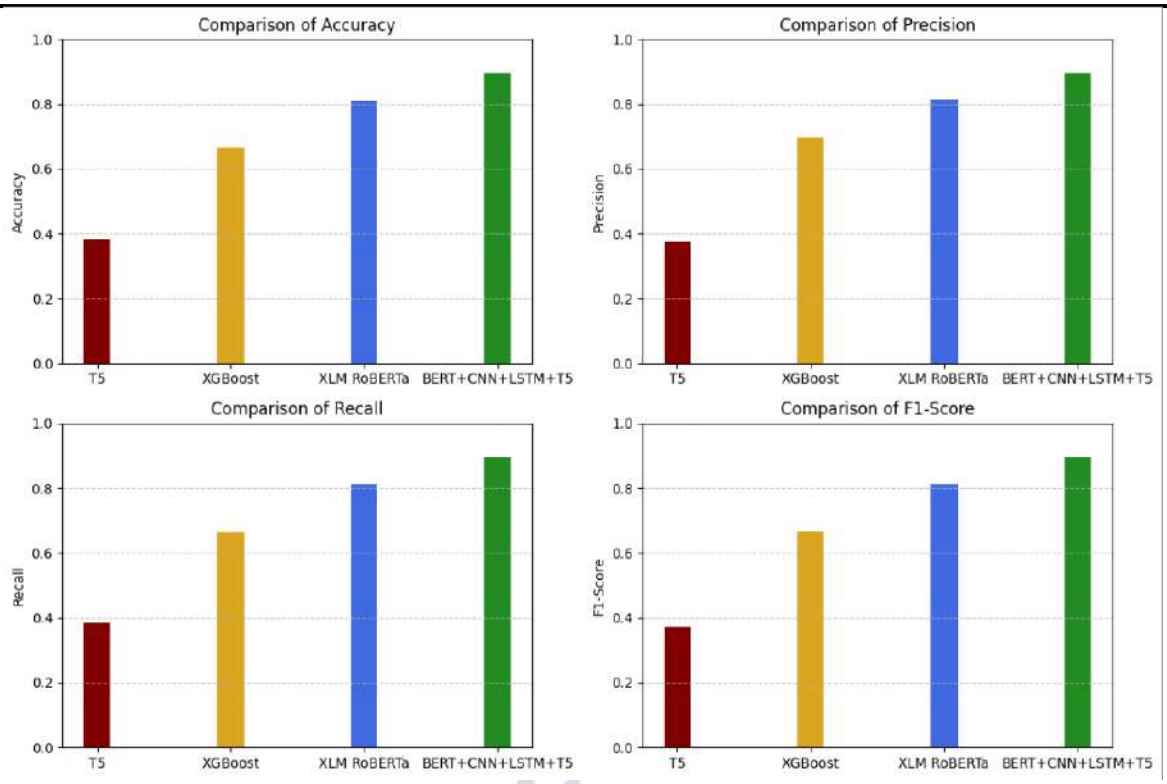


**Figure 17 Sentiment Analysis Comparison**

The hybrid model that combines BERT, CNN, LSTM, and T5 outperformed all other models tested in this paper in terms of all metrics. Its accuracy was 89.56%, precision was 89.65%, recall was 89.56%, and F1-score was 89.58%. Thus, it clearly extracts rich semantic features, models sequential patterns, and classifies effectively. The addition of CNN and LSTM together with transformer-based models such as BERT and T5 had significantly increased capturing both contextual and structural nuances of the Urdu language.

In contrast, XLM RoBERTa also reported good performance, with an accuracy of 81.13%, a precision of 81.30%, a recall of 81.13%, and an F1-score of 81.18%. The model generalized well to the Urdu text since it was pretrained on multilingual data. Still, it missed the domain-specific enhancements from the hybrid approach. XGBoost reported an accuracy of 66.60% and an F1-score of 66.76%. It performed better than T5 but could not capture deeper linguistic patterns since it was heavily dependent on feature engineering and not so sophisticated text representations. T5, on its own, performed the worst, with an accuracy of 38.30% and an F1-score of 37.15%, which reflects that it cannot be used in isolation for sentiment analysis tasks.

**Table 12 Sarcasm Detection Model Comparison**

| | Accuracy | Precision | Recall | F1-Score |
|---|---|---|---|---|
| **T5** | 0.6906 | 0.6496 | 0.7892 | 0.7126 |
| **XGBoost** | 0.7970 | 0.8041 | 0.7701 | 0.7867 |
| **XLM RoBERTa** | 0.8059 | 0.6895 | 0.8031 | 0.7425 |
| **USASD** | **0.8304** | **0.8374** | **0.8304** | **0.8299** |

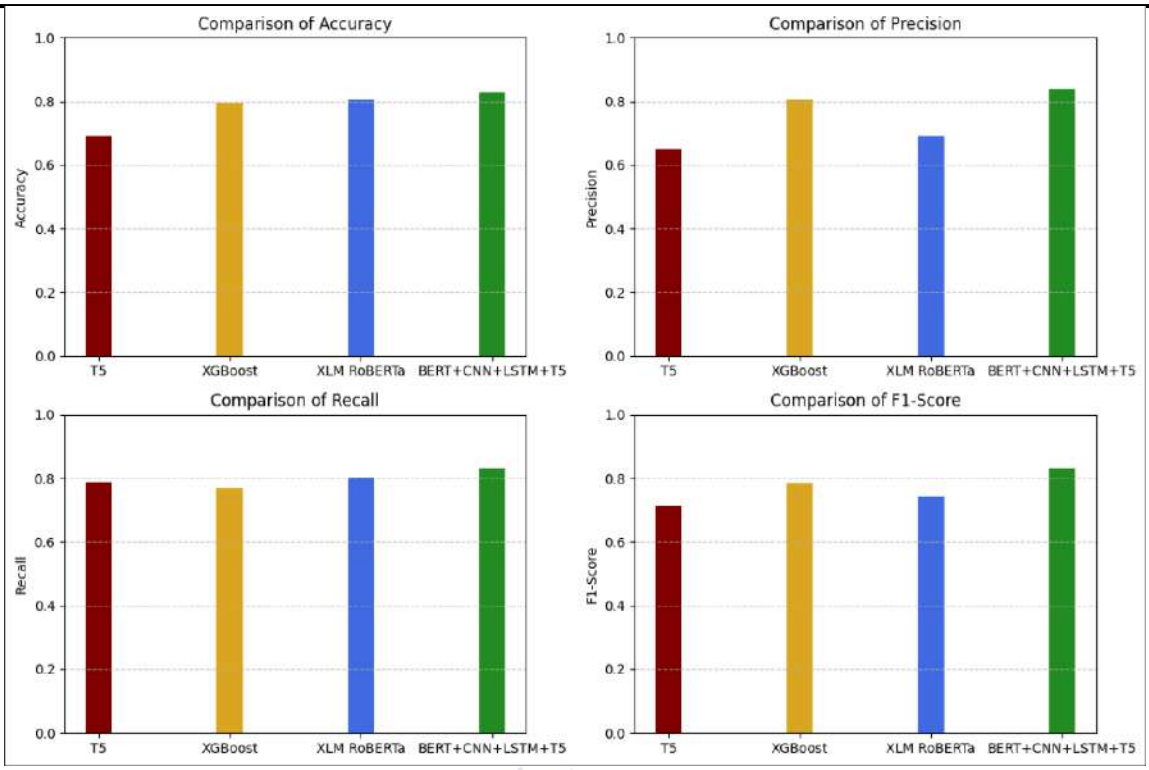


**Figure 18 Sarcasm Detection Comparison**

**Sarcasm Detection Comparison**

The same four models were tested for sarcasm detection. Our USASD model again showed the best performance, capturing the subtle relationship between sentiment and context that is crucial for detecting sarcasm. XLM-RoBERTa, which was trained on multilingual data, also performed well. XGBoost was satisfactory but much worse than the transformer-based models, indicating the weakness of traditional machine learning in capturing the subtlety of sarcasm. T5, as a standalone model, was the weakest performer, and further fine-tuning and architectural improvements are required for sarcasm detection.

For SD, the USASD model again obtained the best performance, which was an accuracy of 83.04%, a precision of 83.74%, a recall of 83.04%, and an F1-score of 82.99%. BERT, CNN, LSTM, and T5 provided a good robust framework to detect the nuances and often ambiguous nature of sarcasm in Urdu text. This was due to the fact that CNN and LSTM are able to capture patterns and sequential dependencies, whereas BERT can understand the context and T5 is able to generate classification capabilities.

XLM RoBERTa achieved 80.59% accuracy and an F1-score of 74.25%. Although very strong, it was not on par with the hybrid model for the specific difficulties of Urdu sarcasm detection, including idiomatic expressions and cultural nuances. XGBoost achieved 79.70% accuracy and an F1-score of 78.67%, showing its ability to handle sarcasm detection up to a reasonable degree but not quite at the same level as deep learning approaches. T5 on its own scored 69.06% accuracy and 71.26% for the F1 score, suggesting modest success but emphasizing that integrating complementary models with this one will be required to achieve state-of-the-art results.

**Conclusion and Future Directions**

This study explored various deep learning models for Urdu sentiment analysis and sarcasm detection, a low-resource language. The models that were compared are T5, XGBoost, XLM-RoBERTa, and USASD, a hybrid BERT+CNN+LSTM+T5 architecture.

Evaluation based on accuracy, precision, recall, and F1-score show that there are considerable performance differences. The USASD model performed the best on all metrics: accuracy (83.04%), precision (83.74%), recall (83.04%), and F1-score (82.99%). It showed that combining feature extraction, contextual understanding, and sequential modeling is effective. XLM-RoBERTa also performed well, especially in recall, but was not able to perform well in precision. XGBoost, being a much simpler model, still delivered competitive results but relies on structured features. T5, although exhibiting high recall, was the weakest performer overall given its lower precision and accuracy, and therefore further tuning or integration with other models are necessary. Looking beyond the challenges of Urdu language intrinsic properties and lack of resources, the research confirms the potential of transformer-based and hybrid architectures for efficient sentiment analysis and sarcasm detection in such languages.

Future research in sentiment analysis and sarcasm detection should explore advanced models like GPT-based architectures and multimodal frameworks. Methods combining multiple advanced models are also promising. Creating larger, more diverse Urdu datasets, incorporating regional dialects and varied contexts, is crucial for improved generalizability. Refining training through advanced loss functions, dynamic learning rates, and data augmentation techniques, including active and semi-supervised learning, can further enhance accuracy. Finally, establishing ethical standards for deployment, addressing bias, and providing clear disclosures and usage guidelines are essential for responsible development.

**Ethical Considerations**

This research is well aware of the ethical considerations that come with the detection of sentiment and sarcasm, especially bias in language models. Though data cleaning and preprocessing were done to reduce bias, vigilance continues to be the watchword. Future implementations will have to prioritize fairness, inclusivity, transparency, and user privacy. Another important thing is to prevent possible misuse by adherence to guidelines and stakeholder awareness.

**REFERENCES**

A, A., G, S., H r, S., Upadhyaya, M., Ray, A. P., & T c, M. (2021). Sarcasm detection in natural language processing. *Materials Today: Proceedings*, *37*, 3324–3331. https://doi.org/10.1016/j.matpr.2020.09.124

Ahmad, W., &Edalati, M. (2022). *Urdu Speech and Text Based Sentiment Analyzer* (arXiv:2207.09163). arXiv. https://doi.org/10.48550/arXiv.2207.09163

Ali, A., Khan, M., Khan, K., Khan, R. U., &Aloraini, A. (2024). Sentiment Analysis of Low-Resource Language Literature Using Data Processing and Deep Learning. *Computers, Materials & Continua*, *79*(1), 713–733. https://doi.org/10.32604/cmc.2024.048712

Alsaeedi, A., & Khan, M. Z. (2019). A Study on Sentiment Analysis Techniques of Twitter Data. *International Journal of Advanced Computer Science and Applications*, *10*(2).

Alshamsi, A., Bayari, R., & Salloum, S. (2020). Sentiment analysis in English texts. *Advances in Science, Technology and Engineering Systems Journal*, *5*(6). https://doi.org/10.25046/aj0506200

Amir, S., Wallace, B. C., Lyu, H., & Silva, P. C. M. J. (2016a). *Modelling Context with User Embeddings for Sarcasm Detection in Social Media* (arXiv:1607.00976). arXiv. https://doi.org/10.48550/arXiv.1607.00976

Amir, S., Wallace, B. C., Lyu, H., & Silva, P. C. M. J. (2016b). *Modelling Context with User Embeddings for Sarcasm Detection in Social Media* (arXiv:1607.00976). arXiv. https://doi.org/10.48550/arXiv.1607.00976

Ashok, D. M., Nidhi Ghanshyam, A., Salim, S. S., Burhanuddin Mazahir, D., & Thakare, B. S. (2020). Sarcasm Detection using Genetic Optimization on LSTM with CNN. *2020 International Conference for Emerging Technology (INCET)*, 1–4. https://doi.org/10.1109/INCET49848.2020.9154090

Bamman, D., & Smith, N. (2015). Contextualized Sarcasm Detection on Twitter. *Proceedings of the International AAAI Conference on Web and Social Media*, *9*(1), Article 1. https://doi.org/10.1609/icwsm.v9i1.14655

Bermingham, A. (2012). *Sentiment analysis and real-time microblog search* [Doctoral, Dublin City University]. https://doras.dcu.ie/16748/

Cambria, E., Schuller, B., Xia, Y., & Havasi, C. (2013). New Avenues in Opinion Mining and Sentiment Analysis. *IEEE Intelligent Systems*, *28*(2), 15–21. IEEE Intelligent Systems. https://doi.org/10.1109/MIS.2013.30

Chowdhary, K. R. (2020). Natural Language Processing. In K. R. Chowdhary (Ed.), *Fundamentals of Artificial Intelligence* (pp. 603–649). Springer India. https://doi.org/10.1007/978-81-322-3972-7_19

Davidov, D., Tsur, O., & Rappoport, A. (2010a). Enhanced Sentiment Learning Using Twitter Hashtags and Smileys. In C.-R. Huang & D. Jurafsky (Eds.), *Coling 2010: Posters* (pp. 241–249). Coling 2010 Organizing Committee. https://aclanthology.org/C10-2028

Davidov, D., Tsur, O., & Rappoport, A. (2010b). Semi-Supervised Recognition of Sarcasm in Twitter and Amazon. In M. Lapata & A. Sarkar (Eds.), *Proceedings of the Fourteenth Conference on Computational Natural Language Learning* (pp. 107–116). Association for Computational Linguistics. https://aclanthology.org/W10-2914

Dhaoui, C., Webster, C. M., & Tan, L. P. (2017). Social media sentiment analysis: Lexicon versus machine learning. *Journal of Consumer Marketing*, *34*(6), 480–488. https://doi.org/10.1108/JCM-03-2017-2141

Dubinsky, Y., Catarci, T., Humayoun, S. R., & Kimani, S. (2007). Integrating user evaluation into software development environments. *2nd DELOS Conference on Digital Libraries, Pisa, Italy*.

Farhan, S., Shoukat, R., & Aslam, A. (2024, May 1). *Automatic Sarcasm Detection on Cross-Platform Social Media Datasets: A GLoVe and Bi-LSTM Based Approach. | EBSCOhost*. https://doi.org/10.3897/jucs.104790

Ghosh, A., & Veale, T. (2016). Fracking Sarcasm using Neural Network. In A. Balahur, E. van der Goot, P. Vossen, & A. Montoyo (Eds.), *Proceedings of the 7th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis* (pp. 161–169). Association for Computational Linguistics. https://doi.org/10.18653/v1/W16-0425

Hassan, M. E., Hussain, M., Maab, I., Habib, U., Khan, M. A., & Masood, A. (2024). Detection of Sarcasm in Urdu Tweets Using Deep Learning and Transformer Based Hybrid Approaches. *IEEE Access*, *12*, 61542–61555. IEEE Access. https://doi.org/10.1109/ACCESS.2024.3393856

Jacquemin, C. (2001). *Spotting and Discovering Terms Through Natural Language Processing*. MIT Press.

Jijkoun, V., de Rijke, M., &Weerkamp, W. (2010). Generating Focused Topic-Specific Sentiment Lexicons. In J. Hajič, S. Carberry, S. Clark, & J. Nivre (Eds.), *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics* (pp. 585–594). Association for Computational Linguistics. https://aclanthology.org/P10-1060

Joshi, A., Bhattacharyya, P., & Carman, M. J. (2017). Automatic Sarcasm Detection: A Survey. *ACM Comput. Surv.*, *50*(5), 73:1-73:22. https://doi.org/10.1145/3124420

Khan, L., Amjad, A., Ashraf, N., Chang, H.-T., &Gelbukh, A. (2021). Urdu sentiment analysis with deep learning methods. *IEEE Access*, *9*, 97803–97812.

Khan, S., Qasim, I., Khan, W., Khan, A., Khan, J. A., Qahmash, A., &Ghadi, Y. Y. (2024). An automated approach to identify sarcasm in low-resource language. *PLOS ONE*, *19*(12), e0307186. https://doi.org/10.1371/journal.pone.0307186

Kharde, V. A., & Sonawane, P. S. (2016). Sentiment Analysis of Twitter Data: A Survey of Techniques. *International Journal of Computer Applications*, *139*(11), 5–15. https://doi.org/10.5120/ijca2016908625

Majeed, A., Mujtaba, H., & Beg, M. O. (2020). Emotion detection in roman urdu text using machine learning. *Proceedings of the 35th IEEE/ACM International Conference on Automated Software Engineering*, 125–130.

Majid, A., Naqvi, U., Shokat, S., & Azeem, A. (2023). A Novel Morphological Rule-based Approach for Urdu Text Sentiment Analysis. *Journal of Information Communication Technologies and Robotic Applications*, *14*(1), Article 1. https://doi.org/10.51239/jictra.v14i1.315

Mandal, P. K., & Mahto, R. (2019). Deep CNN-LSTM with Word Embeddings for News Headline Sarcasm Detection. In S. Latifi (Ed.), *16th International Conference on Information Technology-New Generations (ITNG 2019)* (pp. 495–498). Springer International Publishing. https://doi.org/10.1007/978-3-030-14070-0_69

Mukhtar, N., & Khan, M. A. (2018a). Urdu Sentiment Analysis Using Supervised Machine Learning Approach. *International Journal of Pattern Recognition and Artificial Intelligence*, *32*(02), 1851001. https://doi.org/10.1142/S0218001418510011

Mukhtar, N., & Khan, M. A. (2018b). Urdu sentiment analysis using supervised machine learning approach. *International Journal of Pattern Recognition and Artificial Intelligence*, *32*(02), 1851001.

Mukhtar, N., Khan, M. A., &Chiragh, N. (2018). Lexicon-based approach outperforms Supervised Machine Learning approach for Urdu Sentiment Analysis in multiple domains. *Telematics and Informatics*, *35*(8), 2173–2183. https://doi.org/10.1016/j.tele.2018.08.003

Naqvi, U., Majid, A., & Abbas, S. A. (2021a). UTSA: Urdu text sentiment analysis using deep learning methods. *IEEE Access*, *9*, 114085–114094.

Naqvi, U., Majid, A., & Abbas, S. A. (2021b). UTSA: Urdu Text Sentiment Analysis Using Deep Learning Methods. *IEEE Access*, *9*, 114085–114094. IEEE Access. https://doi.org/10.1109/ACCESS.2021.3104308

Naqvi, U., Majid, A., & Abbas, S. A. (2021c). UTSA: Urdu text sentiment analysis using deep learning methods. *IEEE Access*, *9*, 114085–114094.

Nasukawa, T., & Yi, J. (2003). Sentiment analysis: Capturing favorability using natural language processing. *Proceedings of the 2nd International Conference on Knowledge Capture*, 70–77.

Onan, A. (2017). Sarcasm Identification on Twitter: A Machine Learning Approach. In R. Silhavy, R. Senkerik, Z. Kominkova Oplatkova, Z. Prokopova, & P. Silhavy (Eds.), *Artificial Intelligence Trends in Intelligent Systems* (pp. 374–383). Springer International Publishing. https://doi.org/10.1007/978-3-319-57261-1_37

Qasim, M., Nawaz, S., Hussain, S., & Habib, T. (2016). Urdu speech recognition system for district names of Pakistan: Development, challenges and solutions. *2016 Conference of The Oriental Chapter of International Committee for Coordination and Standardization of Speech Databases and Assessment Techniques (O-COCOSDA)*, 28–32. https://doi.org/10.1109/ICSDA.2016.7918979

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., & Liu, P. J. (2020). Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research*, *21*(140), 1–67.

Rajadesingan, A., Zafarani, R., & Liu, H. (2015). Sarcasm Detection on Twitter: A Behavioral Modeling Approach. *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, 97–106. https://doi.org/10.1145/2684822.2685316

Rehman, E. U., Khan, M. N., & Aziz, N. (n.d.). *Exploring Political Emotions: Sentiment Analysis of Urdu Tweets*. Retrieved January 21, 2025, from https://www.researchgate.net/profile/Ijist-Jr/publication/382398674_Exploring_Political_Emotions_Sentiment_Analysis_of_Urdu_Tweets/links/669b89cc8dca9f441b8a18cd/Exploring-Political-Emotions-Sentiment-Analysis-of-Urdu-Tweets.pdf

Schouten, K., &Frasincar, F. (2016). Survey on Aspect-Level Sentiment Analysis. *IEEE Transactions on Knowledge and Data Engineering*, *28*(3), 813–830. IEEE Transactions on Knowledge and Data Engineering. https://doi.org/10.1109/TKDE.2015.2485209

Shabbir, M., & Majid, M. (2024). Sentiment Analysis From Urdu Language-based Text using Deep Learning Techniques. *2024 5th International Conference on Advancements in Computational Sciences (ICACS)*, 1–5. https://doi.org/10.1109/ICACS60934.2024.10473232

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A., & Potts, C. (2013). Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. In D. Yarowsky, T. Baldwin, A. Korhonen, K. Livescu, & S. Bethard (Eds.), *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing* (pp. 1631–1642). Association for Computational Linguistics. https://aclanthology.org/D13-1170

Syed, A. Z., Aslam, M., & Martinez-Enriquez, A. M. (2011). Sentiment Analysis of Urdu Language: Handling Phrase-Level Negation. In I. Batyrshin& G. Sidorov (Eds.), *Advances in Artificial Intelligence* (pp. 382–393). Springer. https://doi.org/10.1007/978-3-642-25324-9_33

Tahir, B., & Mehmood, M. A. (2022). Anbar: Collection and analysis of a large scale Urdu language Twitter corpus. *Journal of Intelligent & Fuzzy Systems*, *42*(5), 4789–4800. https://doi.org/10.3233/JIFS-219266

Wallace, E., Buil, I., &Chernatony, L. de. (2014). Consumer engagement with self-expressive brands: Brand love and WOM outcomes. *Journal of Product & Brand Management*, *23*(1), 33–42. https://doi.org/10.1108/JPBM-06-2013-0326

Yadollahi, A., Shahraki, A. G., &Zaiane, O. R. (2017). Current State of Text Sentiment Analysis from Opinion to Emotion Mining. *ACM Comput. Surv.*, *50*(2), 25:1-25:33. https://doi.org/10.1145/3057270