Computers, Materials & Continua

Tech Science Press

https://doi.org/10.32604/cmc.2026.079340

ARTICLE

A Streamlined Client-Server Architecture for Sustainable Sentiment Analysis System Using Textual Data

Soumalya De[1], Rahil Akhtar[2], Saiyed Umer[2], Ranjeet Kumar Rout[3] and G. G. Md. Nawaz Ali[4,*]

[1]Department of Information Technology, Techno International New Town, Kolkata, India
[2]Department of Computer Science and Engineering, Aliah University, Kolkata, India
[3]Department of Information Technology, Dr. B. R. Ambedkar National Institute of Technology, Jalandhar, Punjab, India
[4]Department of Computer Science and Information Systems, Bradley University, Peoria, IL, USA

*Corresponding Author: G. G. Md. Nawaz Ali. Email: nali@bradley.edu

Received: 20 January 2026; Accepted: 09 March 2026; Published: 08 May 2026

ABSTRACT: This work presents a comprehensive sustainable sentiment analysis system utilizing textual data, designed within a structured client-server architecture for real-time deployment. The system integrates dual feature representations Bag-of-Words (BoW) and Term Frequency–Inverse Document Frequency (TF-IDF) whose prediction scores are combined through a parameter-free score-level fusion strategy. The implementation of the proposed system consists of five major components. The first component involves the acquisition of textual data from various sources, followed by rigorous text preprocessing to eliminate noise and enhance data quality. The second component focuses on feature extraction, ensuring that the extracted features not only reduce computational overhead but also retain high discriminative capability to effectively represent sentiments. The third component involves the development of sentiment classification models to categorize textual data based on sentiment polarity. The framework is evaluated across binary (2-class), ternary (3-class), and fine-grained (13-class) sentiment and emotion datasets. To address class imbalance in the 13-class setting, LLM-based data augmentation is incorporated during training. Following model development, the fourth component involves performance optimization, where the best-performing feature models are selected, and classifier parameters are fine-tuned to maximize accuracy and efficiency. Finally, in the fifth component, the optimized and scalable sentiment analysis models for both three-class and thirteen-class sentiment categories are deployed to a server environment for sustainable real-time sentiment classification tasks. Experimental results demonstrate that the proposed architecture maintains stable predictive performance while supporting controlled vocabulary sizes for efficient deployment. The system design emphasizes thin-client interaction, stateless REST-based inference, and centralized model management, ensuring practical applicability in real-world environments, making the system suitable for large-scale implementation in various domains such as customer feedback analysis, social media monitoring, and opinion mining.

KEYWORDS: Sustainable; sentiment analysis; emotion recognition; client server; fusion; model deployment

1 Introduction

The ability to extract underlying emotions or sentiments from written text has become an essential application of Natural Language Processing (NLP). This field, known as sentiment analysis, seeks to understand the emotional tone conveyed in a given piece of text or speech [1]. Traditionally, sentiment analysis has classified emotions into three broad categories: positive, neutral, and negative. These categories, however, fail to capture the nuances and complexity of human emotions. This categorization encompassed

a wide range of emotions within a single class. NLP, or natural language processing, deals particularly with artificial intelligence. This process aims to enable the computer to process human-written text data (i.e., natural language) [2]. It is closely related to knowledge representation, information retrieval, and computational linguistics, a branch of linguistics. In client-server deployments, NLP models are essential and sustainable because they ensure seamless user interactions through client apps while effectively managing text processing tasks on the server side [3]. The computational load on client devices is reduced when NLP models are deployed on the server, enabling efficient processing of complex tasks such as entity recognition, text summarisation, and sentiment analysis. This centralized method facilitates smooth updates, makes model maintenance more manageable, and increases scalability by handling several requests simultaneously. Furthermore, the model's structure and data are protected when hosted on the server, enhancing security. Using powerful NLP frameworks like transformers, spaCy, or NLTK further improves speed and delivers quick, precise results suitable for real-time text analysis applications. A basic client-server architecture related to the NLP model is illustrated in Fig. 1.

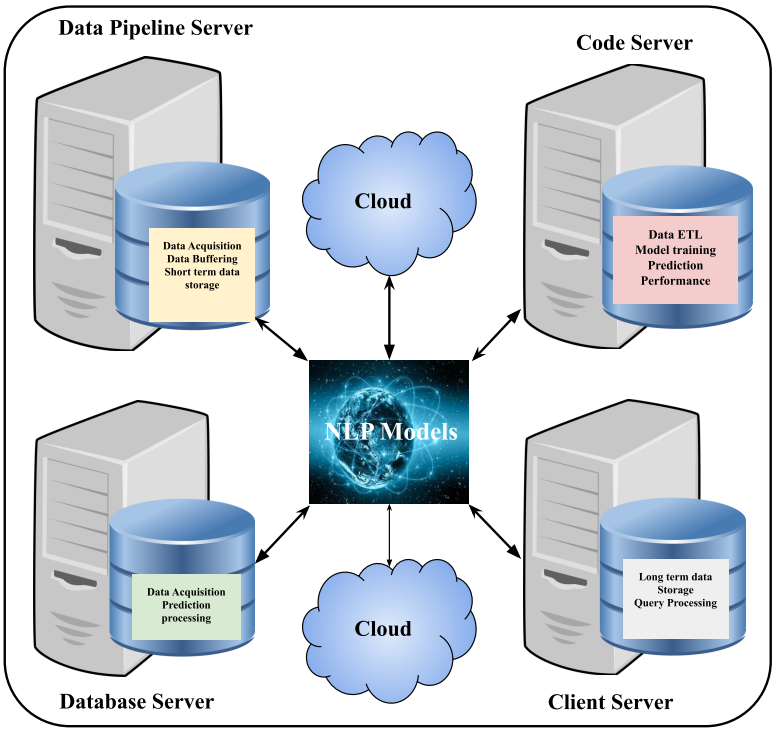


Figure 1: An example of an architectural diagram for the client-server architecture deploying NLP models.

The 'Bag of words' is used for the rule-based sentiment analysis. The Bag-of-Words (BoW) method is one of the most fundamental techniques in sentiment analysis (SA) and natural language processing (NLP). The core idea behind BoW is to represent the words in a sentence as a collection of words, ignoring grammar, syntax, context, and word order, while keeping a record of each word's frequency. The BoW method uses a corpus of annotated words based on the sentiment or emotion the word represents. In this approach, a text corpora is preprocessed to create a vocabulary. A list of all unique words found in the dataset. Following that, individual documents are represented as vectors, with the vector length indicating the vocabulary's size and the value at each point denoting the word frequency in the document. Often called a "word count vector," this representation is used as the feature set for additional analysis, usually for machine learning tasks classification [4]. Text corpora is a dataset of many annotated or non-annotated text data in a digital form.

In sentiment analysis, a text corpus is used to identify patterns of change in word variations and to perform statistical analysis of words to train a machine learning model that predicts the sentiment or emotion of the input text data [5]. For sentiment analysis, 'Bag of words' extracts important terms indicative of a particular sentiment or emotion. For example, words like "happy," "joyful," or "excited" could be strongly associated with positive sentiments, while "sad," "angry," or "frustrated" might be linked to negative sentiments. By analyzing the frequencies of such terms across a given set of documents, BoW models can classify texts into sentiment categories, like positive, negative, or neutral. From the inputted sentence, the words are tokenized, making it easier to process the data for machine learning (i.e. the sentence is broken down into its word components). The Tokenized words may be in adjective, continuous or some other grammatical form. Then, the words are lemmatized; each suffix and prefix of each word is removed, and the word is converted to its lexical root form according to the context of the sentence [6]. Part-of-speech (POS) labelling is an essential stage in lemmatization. Each word tag in the phrase is essentially assigned by part of speech to indicate its contextual meaning.

The processed data is then split into training and test samples. The training samples are used to train the machine learning (ML) model using logistic regression, support vector machines, and random forests. In machine learning and deep learning, data is typically gathered from text corpora using rule-based, statistical, or neural-based methodologies. Natural language Processing (NLP) techniques are used to understand, generate, classify text, and perform speech recognition using computer algorithms and data processing pipelines [7]. Logistic regression is a supervised machine learning algorithm that uses statistical methods for binary classification tasks [8], where the goal is to categorize a given input into one of two possible outcomes (labelled as 0 or 1, true or false). Unlike linear regression, which predicts continuous values, logistic regression outputs a probability that the given input belongs to a particular class. It uses the sigmoid function to transform the linear combination of input variables into a binary value between 0 and 1. This is used to find the decision boundary representing the relationship between the dependent and independent variables and to differentiate the variable classes on a scatter plot. A support vector machine (SVM) is another supervised machine learning algorithm used for classification and regression. However, it is usually used for classification tasks. This classifier finds the optimal hyperplane that separates samples from multiple classes in a high-dimensional space. The objective is to maximize the separation between the hyperplane and the closest support vectors (samples) from either class [9]. A random forest classifier is another supervised learning technique that uses ensemble methods to classify tasks. Multiple decision trees are combined to improve the model's overall accuracy, precision and generalization. The main objective of this approach is to reduce overfitting, which is quite common with decision trees. Averaging the predictions of multiple decision trees reduces overfitting and improves the performance of random forest classifiers. The work in [10] uses AI-based text mining to analyze English and Turkish tweets to investigate how linguistic and cultural variations impact social media conversations around sustainability and green energy. Significant topic variations influenced by language and cultural environment show that negative attitudes predominate in both languages. The Audit Risk Sentiment Value (ARSV), a unique audit risk proxy that uses sentiment analysis to overcome the shortcomings of conventional audit risk indicators, including audit fees, audit hours, and discretionary accruals, is introduced by Wang et al. [11]. Compared to traditional quantitative metrics, ARSV provides a more thorough assessment by capturing subtle aspects of audit risk through qualitative analysis of audit report narratives.

Sentiment analysis is quite good in extracting the straightforward emotions from the sentence; however, the predictions of the sentiment analysis fail when the sentence has a sarcastic tone, uses idioms, meaning negation (Negation in language refers to the process of reversing or denying the meaning of a statement, typically by using words like "not", "no", or "never". It changes an affirmative statement into its opposite, such

as turning “She is happy” into “She is happy, no?”) or slang phrases like “bro cooked”, and “bro is cooked” (“bro cooked” has a positive notion while “bro is cooked” has a negative notion) [12]. This experiment focuses on determining the amounts of positive, negative, and neutral emotions and the nature of those emotions. This experiment describes the more nuanced emotions in 13 classes, namely empty, sadness, enthusiasm, neutral, worry, surprise, love, fun, hate, happiness, boredom, relief and anger from text data [13]. Textual sentiment analysis uses natural language processing techniques, tokenization, lemmatization, and a rule-based approach to train a machine learning model to predict the sentiment of the input text. The approach is that the user will enter a message in a text box. The machine learning model will predict the percentages of positive, negative, and neutral sentiments, and the proportions of those sentiments as emotions. The study in [14] examines the connection between seasonal fluctuations in PM10 pollution levels and media sentiment regarding air pollution. Emotion models are the base of emotion detection. However, the most important emotion detection (ED) models are discrete and dimensional. The discrete emotional model classifies emotions into six distinct classes. Hence, there is no relation among the other emotions in the discrete emotion model. However, the dimensional emotional model assumes that no emotion can be unrelated to any other emotion. Hence, all these classes need to be framed in a two-dimensional space. In the dimensional emotional model, Plutchik [15] presents a two-dimensional wheel of emotion, with valence on the vertical axis and arousal on the horizontal axis. The wheel consists of emotions, whereas the outermost emotions are the derivatives of the eight primary emotions. Fig. 2 shows Plutchik’s Wheel of Emotions, which contains different emotions. The basic emotions are categorized into three layers, and each emotion has a corresponding opposite emotion, represented by opposite petals.

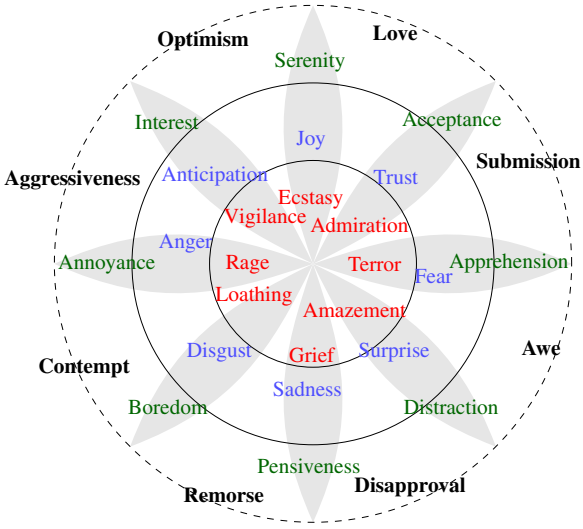


Figure 2: Variations in human emotions illustrated using Plutchik’s wheel of emotions.

Russell [16] presents a two-dimensional circular model where arousal differentiates between activation and deactivation, and valence differentiates between pleasantness and unpleasantness. Fig. 3 depicts Russell’s circumplex model of affect.

The main contributions of this work are as follows:

- Design of a thin-client, server-centric architecture for scalable sentiment and emotion analysis.
- Integration of parallel dual feature streams (BoW and TF-IDF) within a unified deployment framework.
- Implementation of a parameter-free score-level fusion strategy optimized for inference efficiency.
- Memory-aware deployment strategy enabling efficient 13-class fine-grained emotion recognition.

- Empirical validation across 2-class, 3-class, and 13-class datasets with LLM-based augmentation for class balancing.
- The model's effectiveness is demonstrated across multiple sentiment classes, showcasing its ability to reduce deployment costs while improving adaptability to a broader range of emotional categories.

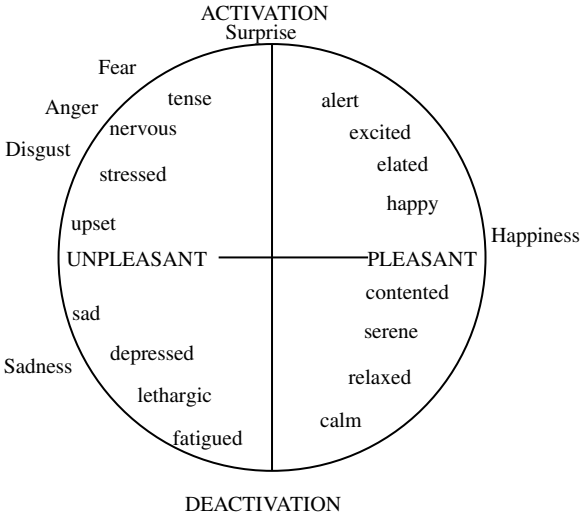


Figure 3: Variations in human emotions illustrated using the two-dimensional circular model of affect.

However, the primary contribution lies in the design and implementation of a scalable client-server architecture that integrates dual feature streams with lightweight score-level fusion for real-time sentiment and emotion analysis. BoW, TF-IDF, and classical supervised learning models are intentionally selected for their low computational overhead. Thus, the novelty is system-level, deployment-focused, and lightweight, with a fusion strategy tailored for sustainable real-time sentiment analysis rather than an algorithmic approach. In addition, we introduce feature-level diversity combined with post-classification score-level fusion, systematically integrating TF-IDF- and BoW-based models within a unified framework and evaluating their effectiveness across 2-class, 3-class, and 13-class emotion settings. Unlike many prior studies limited to binary or coarse-grained sentiment tasks, the proposed framework demonstrates scalable performance for fine-grained emotion recognition, including robustness under class imbalance and LLM-based data augmentation. Finally, the exclusive use of lightweight models and parameter-free fusion rules ensures minimal computational overhead, reinforcing the practical suitability of the approach for real-time, server-side deployment rather than isolated algorithmic performance. The framework proposes an alternative to heavy deep-learning or transformer-based models, rather than a replacement for them, in terms of inference speed, energy efficiency, and maintainability are prioritized over marginal gains in accuracy.

In this article, we are going to organize the paper like: Section 2 refers to the literature reviews and comparative studies in some of the recent research work in the related domain, Section 3 discusses the methodology that we are going to follow to conduct the experiment and Section 4 illustrates the experimental setup, Section 5 follows the result and discussion, followed by conclusion in Section 6 and references at the end.

2 Literature Review

This section reviews relevant literature about sentiment analysis and related methodologies. It is estimated that approximately 80% of the world's data is unstructured and lacks a predefined format,

posing significant challenges in data processing [17]. Managing unstructured data is particularly complex in domains such as natural language processing (NLP), sentiment analysis, and text analytics. Various machine learning algorithms, including Support Vector Machine (SVM), Bayesian Networks (BN), Maximum Entropy (MaxEnt), Conditional Random Fields (CRF), and Artificial Neural Networks (ANN), have been explored to enhance classification accuracy and performance [18]. Recurrent Neural Networks (RNNs) are known to handle sequential textual data effectively; however, their performance tends to degrade when processing lengthy text sequences. To overcome this constraint, the authors of [19] developed an LSTM-based sentiment classification model. Their method involves tokenizing raw textual material, which turns it into integer sequences that are then converted into vector representations through an embedding technique. Following that, the generated vectors are categorized as either positive or negative attitudes. Furthermore, the authors presented an adaptive Emolib classifier for expressive text-to-speech scenarios in [20] in 2013. This model includes the lexical analyzer, sentence splitter, POS tagger, word-sense disambiguation, keyword spotting, stemming, and other text-processing methods. After processing, the model uses classifiers such as Multinomial Naive Bayes (MNB), ARN-R, Latent Semantic Analysis (LSA), Maximum Likelihood Ratio (MLR), and SVM to classify sentiment classes efficiently.

The authors of [21] look at the relationship between human sentiments and emotion detection and suggests a significant change from sentiment analysis to emotional analysis. This change addresses the requirement for greater emotional comprehension across various disciplines. The study highlights that due to marketing strategies, there exists a considerable gap between predictive and diagnostic capabilities in sentiment analysis models. As demonstrated in [22], machine learning models with strong predictive performance may sometimes exhibit weaker diagnostic abilities, mainly when influenced by numerical ratings provided by consumers or users. Several datasets, including ISEAR, SemEval, EmoBank, DailyDialog, Emotion Stimulus, Grounded Emotion, Tweet, and MELD, are widely utilized for emotion detection and sentiment analysis tasks [23]. To improve sentiment extraction from these datasets, the authors in [24] proposed techniques encompassing lexicon-based, machine-learning-based, and deep-learning-based approaches. Notably, a hybrid method combining machine learning algorithms with deep learning techniques has been employed in [25] to improve the accuracy of emotion detection models. Furthermore, a comprehensive review on emotion mining in [26] identifies twelve distinct emotion classes, although psychologists debate the precise categorization of emotions. Ekman's widely adopted six-emotion model remains a standard reference in emotion detection studies. The work by Zucco in [27] provides an insightful overview of various methodologies and available software tools for sentiment analysis. However, as noted in [28], analysing sentiment in lengthy textual data, such as debates or discussions, presents significant challenges, especially when the text exceeds typical model training limits.

Since sentiments are often intertwined with multiple emotions, analyzing extensive textual content requires integrating emotional and opinion-based features [29]. To address this challenge, the authors in [30] proposed a hybrid model combining supervised and unsupervised learning techniques using the SentiWordNet lexicon. Despite its effectiveness, the model's performance accuracy demonstrated variability when applied to identical textual data under different contextual conditions. A detailed comparative analysis of various sentiment analysis models is presented in Table 1. The choice of a 13-class emotion taxonomy in this work is motivated by prior evidence that fine-grained emotion representations better capture the diversity of human affect than a small set of basic emotions. Recent studies have demonstrated the effectiveness of large emotion inventories, including 26 emotion categories in natural image-based emotion recognition [31], 27 distinct emotions identified through large-scale behavioral analysis [32], and 28 emotion classes for modeling complex and mixed affective states in sentiment analysis tasks [33]. In this context, the 13-class taxonomy adopted in this study is not proposed as a new psychological theory, but is inherited from a widely used,

publicly available benchmark dataset commonly employed in computational sentiment and emotion analysis research. The primary objective of this work is to design and evaluate a deployable, scalable sentiment analysis system; therefore, the emotion granularity is application-driven and data-oriented rather than theoretically prescriptive. Importantly, the selected emotion categories remain consistent with established emotion models: several classes (e.g., anger, sadness, happiness, and surprise) directly correspond to Ekman's basic emotions, others (e.g., love, worry, and relief) align with compound emotions in Plutchik's wheel, and low-arousal or neutral states (e.g., empty, boredom, and neutral) are well captured within dimensional models such as Russell's circumplex. This grounding situates the proposed taxonomy within established affective theory while supporting practical, fine-grained emotion recognition.

Table 1: Comparative study on recent research paper on SA.

Sl No.	Paper Title	Model Used	Methodology	Database Used	Drawback
1	Text-Based Sentiment Analysis Using LSTM	LSTM	The model processes raw text input by converting text into integers using a tokenizer. It employs embedding techniques to transform these integers into real-valued vectors. The text sequence is processed using LSTM, which subsequently classifies the text as positive or negative.	IMDB movie reviews, Amazon product reviews	Limited robustness in the embedding system.
2	Sentence-Based Sentiment Analysis for Expressive Text-to-Speech	Emolib Processing Pipeline	Emolib utilizes a lexical analyzer, sentence splitter, POS tagger, word-sense disambiguator, keyword spotter, and stemmer to process raw text. After processing, it employs classifiers such as MNB, ARN-R, LSA, MLR, and SVM for sentiment classification.	SemEval 2007 Dataset, Twitter Dataset	Computational limitations due to the large size of training data.
3	Text-Based Emotion Recognition Using a Deep Learning Approach	Hybrid (CNN + Bi-GRU + SVM)	The model represents text as vectors and processes them using a CNN model. To address the vanishing gradient issue, a bidirectional GRU is employed. Finally, an SVM classifier predicts emotions across six distinct classes.	ISEAR, WASSA, Emotion Stimulus	Lacks real-time emotion detection and ensemble techniques, resulting in increased computation time.

(Continued)

Table 1 (continued)

Sl No.	Paper Title	Model Used	Methodology	Database Used	Drawback
4	A Model for Sentiment and Emotion Analysis of Unstructured Social Media Text	Supervised and Unsupervised Learning Using SentiWordNet Lexicon	The model combines supervised and unsupervised learning techniques, utilizing the SentiWordNet lexicon and corpus-specific lexicons for classification.	Emotion, SMS	The accuracy of the model varies depending on the performance of the SentiWordNet lexicon.

3 Proposed Methodology

This paper proposes a diversified sentiment analysis model capable of classifying textual sentiments. The model is initially designed to categorize sentiments into three distinct classes: positive, negative, and neutral, operating as a stand-alone system. Subsequently, the model is intended for deployment on a server to facilitate real-time sentiment classification from textual data. The system's operational framework is illustrated in Fig. 4, with each component explained in detail in the subsequent subsections. Then, the further proposed model is extended to thirteen classes of emotional sentiments such as "Sadness", "Empty", "Enthusiasm", "Neutral", "Worry", "Surprise", "Love", "Fun", "Hate", "Happiness", "Boredom", "Relief", and "Anger".

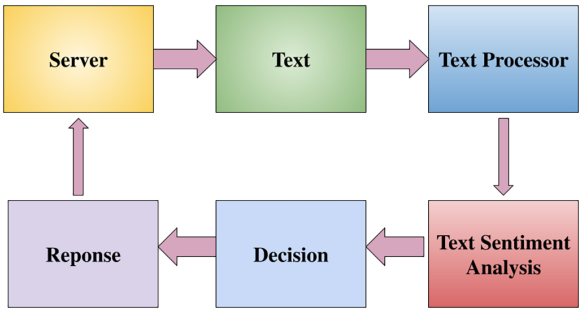


Figure 4: Block diagram of the proposed system.

3.1 Text Processing

The initial stage of the proposed system focuses on text preprocessing to enhance the efficiency of sentiment classification. Each sample (i.e., sentence) from the dataset is first converted to a string and then converted to lowercase. The sentence is subsequently tokenized, dividing it into individual words. Following this, common English stopwords, such as articles (e.g., *was*, *a*, *the*, *were*, etc.) and other non-essential terms that do not significantly impact the sentence's meaning are removed using list comprehension. This process produces a refined token list that retains only the essential words conveying the sentence's core meaning. Every word in the filtered list is then lemmatized, transforming it into its base or root form while preserving

contextual relevance. Lowercase text, filtered words, and lemmatized phrases make up the resultant tokens combined to form coherent sentences. Following processing, these sentences are saved in a specific dataset and used to train the machine learning model. Fig. 5 illustrates the general workflow of the text preparation phases used in the proposed system.

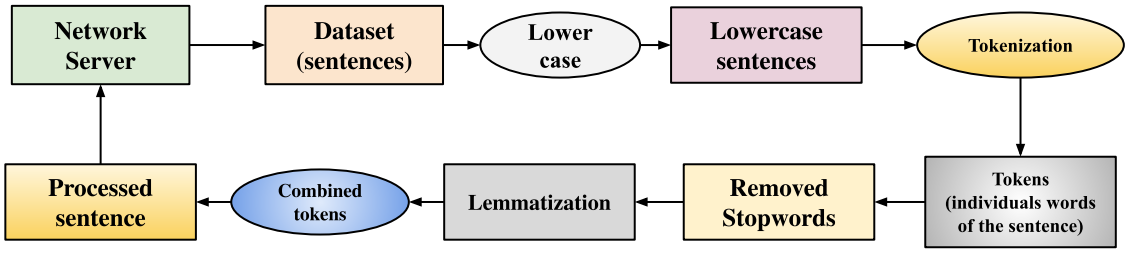


Figure 5: Working flow diagram of text processing employed in the proposed system.

3.2 Feature Extraction Schemes

Raw textual data is inherently unstructured and should be transformed into numerical representations before being processed by machine learning models. In this work, we incorporated two lightweight and deployment-efficient vectorization schemes: Bag-of-Words (BoW) [34] and Term Frequency-Inverse Document Frequency (TF-IDF) [35,36]. The selection of these representations is based on architectural and deployment considerations rather than algorithmic novelty. Both techniques are computationally efficient, memory-controllable, and well-suited for scalable server-side inference. Within the proposed framework, each representation forms an independent feature stream, enabling parallel prediction and subsequent score-level fusion.

(A) TF-IDF feature representation scheme (f_{TFIDF}): TF-IDF assigns importance to terms based on their frequency within a document and their distribution across the corpus. For a term s in document z, the term frequency is defined as:

$$\mathrm{TF}(s,z) = \frac{\text{Count of word } s \text{ in document } z}{\text{Total number of words in document } z}$$

The inverse document frequency over corpus $\mathcal{Q}$ is computed as:

$$\mathrm{IDF}(s,\mathcal{Q}) = \log\left(\frac{|\mathcal{Q}|}{|\{z \in \mathcal{Q} : s \in z\}|}\right) + 1$$

The combined TF-IDF score is:

$$\text{TF-IDF}(s,z,\mathcal{Q}) = \left(\frac{\text{Frequency of word } s \text{ in document } z}{\text{Total words in document } z}\right) \log\left(\frac{|\mathcal{Q}|}{|\{z \in \mathcal{Q} : s \in z\}|}\right) + 1$$

This produces a feature vector $f_{TFIDF} \in \mathbb{R}^{1\times p}$ for each document. Vocabulary size p is controlled during training to maintain deployment feasibility and memory efficiency.

Lemmatization: Lemmatization is applied during preprocessing to reduce morphological variations while preserving semantic meaning. Formally,

$$F : \mathcal{S} \times POS \to \mathcal{L}, \quad F(s,p) = l$$

where $\mathcal{S}$ is the token set, *POS* represents part-of-speech tags, and $\mathcal{L}$ denotes the set of lemmas. This preprocessing step improves feature consistency without increasing dimensionality.

(B) BoW feature representation scheme (f_{BoW}): The BoW representation encodes term occurrence counts without considering word order. For a vocabulary $\mathcal{V}$, a document z is represented as:

$$\text{BoW}(z) = [f_1, f_2, \ldots, f_n]$$

where f_i denotes the frequency of the i-th vocabulary term in z. Similar to TF-IDF, vocabulary size is constrained to ensure scalable deployment.

(C) Sentence-based feature computation: Each sentence s is transformed into a feature vector using either f_{TFIDF} or f_{BoW}. These vectors serve as inputs to their respective classifiers, generating prediction scores. The two independent streams enhance representational diversity while maintaining low computational overhead.

(D) Paragraph-based feature computation: For a paragraph containing n sentences, features are computed as:

$$ft_i = F(s_i), \quad i \in \{1, \ldots, n\}$$

Each feature vector ft_i is evaluated by classifier C:

$$sc_i = C(ft_i)$$

Scores are ranked in non-increasing order:

$$sc_{r_1} \geq sc_{r_2} \geq \cdots \geq sc_{r_n}$$

The paragraph-level prediction $\hat{y}$ is determined using:

$$\hat{y} = \arg\max_j \sum_{i=1}^{k} \mathbb{I}(sc_{r_i} = j)$$

This aggregation strategy prioritizes highly confident sentence-level predictions while maintaining computational efficiency within the server environment.

3.3 Classification Approaches

The classification procedure is about arranging the input sentences into sentiment categories, which will give a sentiment class $\mathcal{C}$ and thereafter transform these sentiments into emotion classes $\mathcal{E}$. The sentiment class $\mathcal{C}$ is a parent sentiment class containing the sub-emotion classes $\mathcal{E}$. Various machine learning classifiers are used for this purpose.

* **Logistic Regression:** This method of supervised learning forecasts a dependent variable which is y, from an independent variable x. The sigmoid function computes the probability that an independent variable x belongs to a class.
* **Support Vector Machine (SVM):** The SVM constructs a hyperplane to predict the classes of the target variable. It offers advantages when the dataset is large and has high dimensions regarding the number of features. The form of the decision function can be written as: $f(\boldsymbol{x}) = \text{sign}\left(\mathbf{w}^T\boldsymbol{x} + b\right)$.
* **Random Forest Classifier:** This method combines many decision trees to eliminate issues with low stability and high overfitting. Each tree T_m is built using a random sample of the data $D_m \subset D$, and these

trees are combined into a dataset D, which is the complete dataset. In most cases, combining systems by majority voting is more favourable for reaching the output.

3.4 Architecture Overview

The architecture is designed to minimize client-side computation, ensure scalability and sustainability under latency benchmarking, multiple concurrent requests, and enable efficient model maintenance by itegrating dual feature representation through centralized deployment. The proposed framework adopts a lightweight client-server architecture designed for real-time sentiment and emotion analysis under constrained computational resources.

3.4.1 Design Principles

The architecture is guided by the following deployment-oriented principles:

- **Thin-client design:** We incorporated all computationally intensive operations like preprocessing, feature extraction, classification, and fusion exclusively on the server. This ensures reduced client-side resource requirements for input acquisition and visualization without requiring modifications.
- **Dual-stream feature processing:** Raw input is transformed into two independent feature representations (f_{TFIDF} and f_{BoW}), enabling complementary modeling without increasing architectural complexity.
- **Parallel inference:** Each feature stream is processed independently by its respective trained classifier, allowing modular scalability and controlled computation overhead.
- **Stateless REST-based inference mechanism:** The architecture employs a RESTful API to handle prediction requests. Each request is processed independently without retaining session-specific state information. This stateless design enhances horizontal scalability, simplifies load balancing, and supports distributed deployment environments.
- **Parameter-free fusion:** Score-level fusion is applied to combine classifier outputs without introducing additional trainable components, ensuring minimal inference overhead.
- **Memory-aware model serialization and reuse:** All trained classifiers and vectorizers are serialized after the offline training phase and loaded into memory during server initialization. Vocabulary sizes are explicitly controlled to limit memory consumption, particularly for the 13-class emotion configuration. This approach ensures predictable resource usage and stable runtime performance.

3.4.2 System Workflow

At first the client submits raw textual input to the server via a REST-based API. Applying preprocessing steps (tokenization, normalization, lemmatization) in the server side, feature vectors are generated using serialized vectorizers trained during the offline phase. The independent prediction scores which are produced from both feature streams undergoes score-level fusion to determine the final sentiment label. Finally the predicted label is returned to the client interface. The model inference is represented in Algorithm 1.

Algorithm 1: Server-side sentiment inference with score-level fusion

Require: Text query T received from client
Ensure: Final sentiment/emotion label $\hat{y}$
1: Receive text input T via API request
2: Preprocess T:

(Continued)

Algorithm 1 (continued)

lowercase conversion, tokenization, stopword removal, lemmatization
Generate feature representations:
$f_{\text{TFIDF}} \leftarrow \text{TFIDF\_Vectorizer}(T)$
$f_{\text{BoW}} \leftarrow \text{BoW\_Vectorizer}(T)$
Perform model inference:
$s_{\text{TFIDF}} \leftarrow \text{Classifier}_{\text{TFIDF}}(f_{\text{TFIDF}})$
$s_{\text{BoW}} \leftarrow \text{Classifier}_{\text{BoW}}(f_{\text{BoW}})$
Fuse classification scores:
$s_{\text{fused}} \leftarrow \text{FusionRule}(s_{\text{TFIDF}}, s_{\text{BoW}})$
Predict final label:
$\hat{y} \leftarrow \arg\max(s_{\text{fused}})$
Return $\hat{y}$ to client

All preprocessing modules and trained models are serialized and reused during inference to ensure consistency between training and deployment environments.

3.4.3 Modular Deployment Structure

The architecture is organized into the following deployable components:

- **Frontend Layer:** Provides user interaction through web-based input forms.
- **Application Layer:** Handles API routing, preprocessing, feature transformation, and prediction logic.
- **Model Layer:** Contains serialized classifiers and vectorizers.
- **Fusion Layer:** Implements score-level aggregation without additional learning parameters.

This modular separation enables independent updating of models without modifying client-side components, improving maintainability and long-term sustainability.

3.4.4 Scalability across Class Configurations

The architecture supports 2-class, 3-class, and 13-class sentiment configurations. While binary and ternary setups are computationally lightweight, the 13-class setting requires careful control of feature dimensionality to ensure deployment feasibility. The considered sentiment classes for 13-class settings are: Empty, Sadness, Enthusiasm, Neutral, Worry, Surprise, Love, Fun, Hate, Happiness, Boredom, Relief, and Anger. To evaluate scalability, models are trained using feature sizes of 5000, 4000, 3000, 2000, 1000, and 500 terms. This analysis quantifies the trade-off between memory consumption and predictive performance for fine-grained emotion recognition.

3.4.5 Deployment Efficiency Considerations

During deployment, the trained classifiers and corresponding vectorizers (f_{TFIDF} and f_{BoW}) are serialized and hosted on the server. For each incoming request:

1. Raw text is received via API.
2. Preprocessing (tokenization, lemmatization, stopword removal) is applied.
3. Feature vectors are generated using the stored vectorizers.
4. Independent model predictions are computed.
5. Score-level fusion determines the final sentiment label.

A minimum server configuration of 1 GB RAM is sufficient to handle inference for all class settings. Dependency consistency is maintained using the `requirements.txt` configuration to ensure reproducible deployment. Algorithm 2 outlines the deployment workflow.

Algorithm 2: Model deployment for sentiment analysis

Require: Trained sentiment analysis model, feature vectorizer, frontend, and backend files
Ensure: Deployed model accessible via a domain/subdomain

1: Map domain/subdomain to server IP.
2: Upload model, vectorizer, frontend, and backend files.
3: Configure Python application environment.
4: Install dependencies.
5: Load serialized models in backend.
6: Process user input and generate prediction.
7: Fused prediction to frontend
8: **return**

The proposed model ensures practical deployment feasibility by incorporating controlled vocabulary sizes to limit memory footprint, pre-serialized models to eliminate runtime retraining, minimal dependency overhead and deterministic inference workflow. Sustainability is achieved through thin-client design, parameter-free score-level fusion, and stateless API-based inference for horizontal scalability. The resulting system achieves low-latency inference while maintaining robustness across varying sentiment granularities. The architectural design thus prioritizes operational sustainability, maintainability, and scalability in real-world environments. Fig. 6 illustrates the overall workflow of the system.

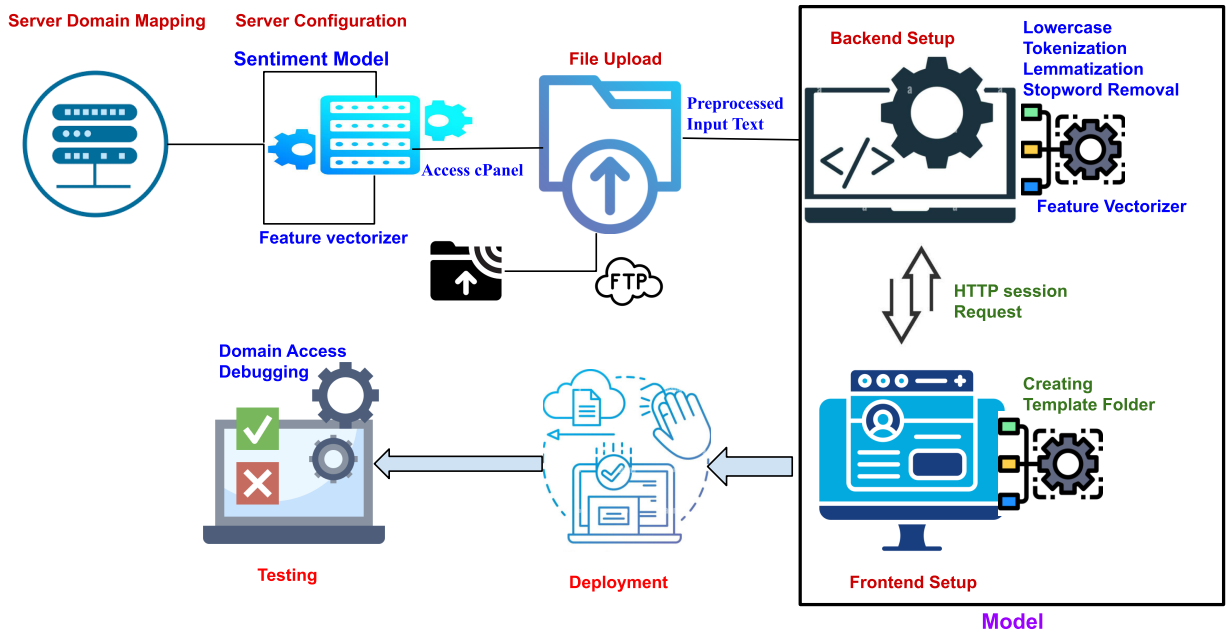


Figure 6: The proposed client server architectural designed for deploying textual sentiment analysis models.

4 The Experiment

This section outlines the experiments, dataset preprocessing, data augmentation, performance with respect to multiple ML classifier, results and discussion pertaining to the proposed model. The experimental evaluation is conducted using two datasets. Initially, the total number of samples in each sentiment class is determined, followed by an equal division of samples within each class. Based on this partitioning, three datasets, denoted as $\mathcal{Z}_1$, $\mathcal{Z}_2$, and $\mathcal{Z}_3$, are created, each comprising 50% of the samples from every sentiment class. These datasets are used as the training and test sets. The proposed sentiment analysis system is developed in Python and runs on an Ubuntu operating system with 32 GB RAM and an Intel Core i7 processor (3.20 GHz). The implementation uses various Python libraries, including the Natural Language Toolkit (NLTK) and machine learning tools from scikit-learn, such as SVM, SVC, RandomForestClassifier, LogisticRegression, and TfidfVectorizer, as well as multiple evaluation metrics. Machine learning models are trained using a 50-50 dataset split, and sentiment classification is performed based on written text. The following subsections describe the datasets used and how augmentation has been applied for training the model.

4.1 Dataset $\mathcal{Z}_1$: Binary Sentiment Classification (2-Class)

The first dataset, denoted as $\mathcal{Z}_1$, is used for binary sentiment classification and contains 150,001 tweet samples annotated with two sentiment labels: positive and negative. This dataset is sourced from a publicly available Kaggle repository commonly used for sentiment analysis benchmarking. The dataset distribution comprises 79,085 positive and 70,916 negative samples. For experimental evaluation, the dataset is randomly partitioned into 50:50 train-test splits, yielding 75,000 samples per subset while preserving class balance. This dataset is primarily used to evaluate baseline sentiment classification performance and classifier robustness in a low-granularity setting. The dataset is obtained from the tweet global warming, a subset of the Crowdflower dataset. Link to dataset: https://huggingface.co/datasets/tasksource/crowdflower/viewer/tweet%20global%20warming. The number of samples from dataset $\mathcal{Z}_1$ is presented in Table 2.

Table 2: The Dataset 1 ($\mathcal{Z}_1$) has annotated data of tweets labelled with two classes. Positive and negative are shown.

Item	Data Quantity	50% of Data
Total Number of Data	150,001	75,000
Label 1 (positive)	79,085	39,542
Label 0 (negative)	70,916	35,458

4.2 Dataset $\mathcal{Z}_2$: Ternary Sentiment Classification (3-Class)

The second dataset, referred to as $\mathcal{Z}_2$, is designed for three-class sentiment classification with the labels positive, negative, and neutral. This dataset is compiled from standard sentiment analysis teaching resources and lecture materials that are widely used in academic research and instructional settings. The dataset contains 27,481 samples per class, ensuring a balanced class distribution. Similar to Dataset $\mathcal{Z}_1$, a 50:50 split is applied to divide the data into training and testing subsets. Dataset $\mathcal{Z}_2$ enables evaluation of the proposed system's ability to distinguish neutral sentiment in addition to positive and negative polarities, which is more challenging than binary classification. This dataset is obtained from the subset of CrowdFlower dataset on airline-sentiment from HuggingFace. Link to dataset: https://huggingface.co/datasets/tasksource/crowdflower/viewer/airline-sentiment. The number of samples from dataset $\mathcal{Z}_2$ is shown in Table 3.

Table 3: The Dataset 2 ($\mathcal{Z}_2$) has annotated data of tweets labelled with three classes. Positive, negative and neutral are shown.

Class	Samples	50% of samples	Total samples
Positive class	8582	4291	27,481
Negative class	7781	3890	27,481
Neutral labeled data	11,118	5559	27,481

4.3 Dataset $\mathcal{Z}_3$: Fine-Grained Emotion Classification (13-Class)

The third dataset, denoted as $\mathcal{Z}_3$, supports fine-grained sentiment and emotion recognition across 13 distinct emotion categories: empty, sadness, enthusiasm, neutral, worry, surprise, love, fun, hate, happiness, boredom, relief, and anger. This dataset is obtained from the CrowdFlower emotion dataset on HuggingFace, a widely used benchmark in emotion-aware sentiment analysis. Initially, the dataset exhibited significant class imbalance, with underrepresented classes such as empty, enthusiasm, boredom, and anger. To mitigate this issue and improve model generalization, synthetic data augmentation was performed by generating additional samples for minority classes using a generative large language model (LLaMA 3:8B). After augmentation, each emotion class was standardized to 44,000 samples, resulting in a balanced dataset. As with the previous datasets, a 50:50 train-test split was employed. Dataset $\mathcal{Z}_3$ is used to evaluate the scalability and robustness of the proposed model under high-granularity emotion classification scenarios. Link to dataset: https://huggingface.co/datasets/tasksource/crowdflower/viewer/textemotion?views%5B%5D=textemotion. It was observed that certain classes, specifically empty, anger, enthusiasm, and boredom, had significantly fewer samples compared to other classes. Such class imbalance can negatively impact the model's ability to learn generalizable patterns, potentially leading to biased predictions. To mitigate this issue and ensure a more balanced representation of all sentiment categories, 1000 additional samples were synthetically generated for each underrepresented class using generative AI. Llama 3:8B was the model utilized to augment the data. Table 4 summarizes the final distribution of samples in $\mathcal{Z}_3$.

Table 4: Emotion data distribution ($\mathcal{Z}_3$).

Class	Data Quantity	50% of Samples	Total Samples
Sadness	5165	2582	44,000
Empty	1827	913	44,000
Enthusiasm	1759	879	44,000
Neutral	8638	4319	44,000
Worry	8459	4229	44,000
Surprise	2187	1093	44,000
Love	3842	1921	44,000
Fun	1776	888	44,000
Hate	1323	661	44,000
Happiness	5209	2604	44,000
Boredom	1179	589	44,000
Relief	1526	763	44,000
Anger	1110	555	44,000

4.4 LLM Based Data Augmentation

To address class imbalance in the 13-class emotion dataset, we employ LLaMA-3 (8B), an open-weight, decoder-only transformer language model developed by Meta, as a prompt-driven text generation module for class-conditional data augmentation. The moderate model size enables scalable synthetic data generation with low computational overhead, supporting reproducibility in server-side sentiment analysis pipelines. Synthetic samples are generated for underrepresented emotion classes (empty, anger, enthusiasm, and boredom) using structured class-conditional prompts that explicitly specify the target emotion, first-person narration, informal social-media style, and sentence-length constraints. Temperature controls the randomness of token selection. Low temperature generates more deterministic and repetitive output while high temperature generates more diverse, creative outputs. To maintain the balance between these two we have selected a balanced temperature of 0.7. The generated tokens need sampling, so instead of sampling all tokens, we focused on a balanced yet diversified sampling. To achieve this, we set top-p to 0.9. Thus, text generation is performed with fixed decoding parameters to ensure consistency across classes: temperature = 0.7, top-p = 0.9, and maximum tokens = 40, using nucleus sampling. These settings balance lexical diversity and semantic coherence while avoiding overly deterministic or noisy outputs. To ensure data quality, an automatic filtering step removes duplicate and semantically irrelevant samples, followed by manual inspection of a randomly selected subset. All synthetic data are used exclusively for training to mitigate class imbalance and are strictly excluded from validation and test sets. Model evaluation is conducted solely on human-annotated samples, ensuring unbiased performance assessment. Table 5 shows the example of generated samples to reduce class imbalance. The method used to generate and validate this augmentation is as follows in the Algorithm 3.

Table 5: Examples of annotated data using LLama-3 8B.

Emotion Class	Example Synthetic Sentences
Empty	(1) "Just staring at the screen with nothing really going on in my head."
	(2) "Today feels blank, like I'm moving but not feeling anything."
Anger	(1) "I'm so tired of being ignored, this is absolutely frustrating."
	(2) "Nothing worked today and honestly I'm furious about it."
Enthusiasm	(1) "I can't wait for tomorrow, everything feels exciting right now!"
	(2) "This project is finally coming together and I'm loving every bit of it."
Boredom	(1) "I've been scrolling for hours and there's still nothing interesting."
	(2) "This day is dragging on forever with absolutely nothing to do."

Algorithm 3: LLM-based data augmentation with quality control

Require: Original training dataset D, underrepresented emotion classes C_u, LLM model M (LLaMA-3, 8B), target sample count N

Ensure: Augmented training dataset D'

1: Initialize $D' \leftarrow D$
2: **for** each emotion class $c \in C_u$ **do**
3: Define class-conditional prompt P_c specifying:
4: target emotion label c, informal style, first-person narration, sentence length constraint
5: Generate N candidate samples using M with fixed parameters:

(Continued)

Algorithm 3 (continued)

temperature = 0.7, top-p = 0.9, max tokens = 40, nucleus sampling
Store generated samples in temporary set G_c
end for
Quality Control:
for each G_c **do**
Remove duplicate and near-duplicate samples using cosine similarity on TF-IDF vectors
Discard incomplete, off-topic, or emotionally ambiguous samples
Perform manual inspection on a random subset of retained samples
Add validated samples from G_c to D' with label c
end for
return D'

4.5 Statistical Robustness and Evaluation Stability

To address statistical robustness, we conducted multiple experimental runs with different random initializations while preserving the same train-test split. For each task (2-class, 3-class, and 13-class), experiments were repeated across multiple independent runs, and performance metrics are reported as: Mean ± Standard Deviation. This approach captures both central performance tendency and variability arising from stochastic elements such as model initialization and training dynamics. The observed standard deviation across repeated runs remains low, indicating stable convergence behavior and consistent generalization performance. These findings confirm that the reported improvements, especially those obtained via score-level Product fusion, are not example of a single experimental instance but demonstrate reproducible and statistically stable behavior.

4.6 Effect of Performance with Multiple ML Classifiers

Here experiment starts with considering 2-class sentiment Dataset $\mathcal{Z}_1$, where randomly 4000 text samples have been considered to check the performance of the proposed system with multiple machine learning classifiers. These performance have been reported in Table 6. From this Table it has been observed that the performance metrics of various machine learning models evaluated on a binary sentiment classification task. The metrics considered include Accuracy, Precision, F1 Score, Receiver Operating Characteristic (ROC) curve, Area Under the Curve (AUC), and Testing Time (in seconds). The models compared are Logistic Regression, Random Forest, Support Vector Machine (SVM), K-Nearest Neighbors, Naïve Bayes, Gradient Boosting, Decision Tree, and AdaBoost. Among these, SVM achieves the highest accuracy (87.55%) and F1 score (87.47), demonstrating strong performance in distinguishing between the two sentiment classes. Logistic Regression also performs competitively with an accuracy of 86.50% and a comparable precision of 86.96. In contrast, K-Nearest Neighbors shows the lowest F1 Score (47.60), suggesting that it struggles to generalize well for this dataset. Moreover, the ROC AUC scores reinforce these findings, with SVM (94.07) and Logistic Regression (94.06) exhibiting the best classification capability, whereas K-Nearest Neighbors has the lowest ROC AUC of 61.11.

The distribution of classes in the training and testing datasets is also shown in the Table 6. There are 925 positive and 1075 negative samples in the training set and 930 positive and 1070 negative samples in the testing set. The models can effectively generalize because of the nearly balanced distribution, which ensures that there is no discernible bias toward any specific class. SVM requires the longest inference time (0.41 s), whereas Logistic Regression, Naïve Bayes, and Decision Trees exhibit nearly instantaneous execution times. The testing time demonstrates the computational efficiency of each model. This is an

essential factor for real-time applications where inference speed is a top concern. Overall, the findings point to SVM and Logistic Regression as the best sentiment classification models because they balance computational efficiency and predictive performance. Logistic Regression, Random Forest, and Support Vector Machine (SVM) were chosen for additional experiments based on their superior performance across important evaluation metrics. Computational efficiency is another important factor affecting this choice. Even though SVM has the longest inference time (0.41 s), its higher predictive capacity makes it worthy of being used in additional research. Conversely, logistic regression runs nearly instantly, which makes it perfect for real-time applications. Random Forest provides a useful trade-off between complexity and efficiency by balancing execution speed and performance. Furthermore, these models—Random Forest, an ensemble-based technique; SVM, a kernel-based classifier; and Logistic Regression, a linear model—represent a variety of learning methodologies. Because of this diversity, a thorough investigation of various machine learning paradigms is ensured, enabling additional development and optimization by the particular demands of the sentiment analysis task.

Table 6: Performance metrics and class distribution with multiple machine learning classifiers applied on dataset $\mathcal{Z}_1$.

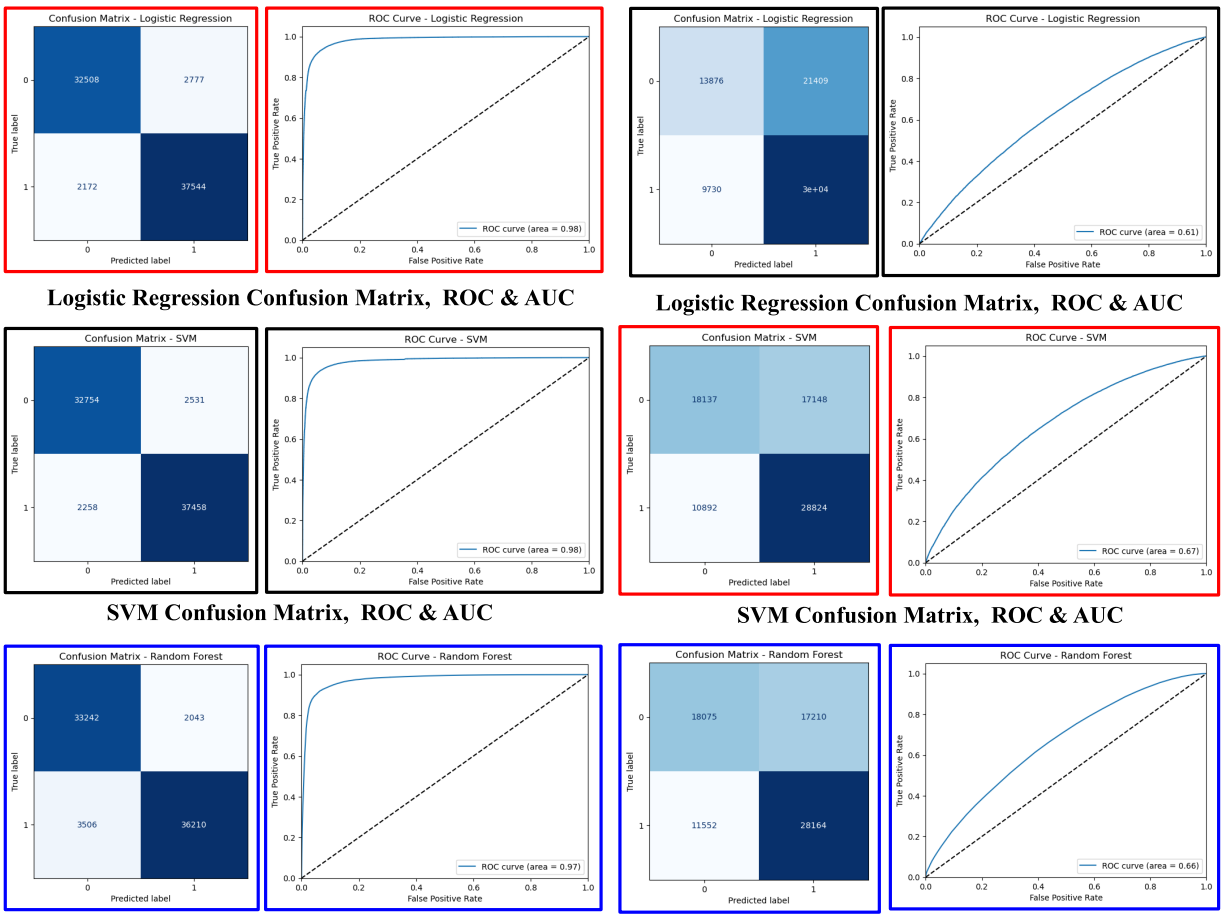
Model	Accuracy	Precision	F1-Score	ROC AUC	Testing Time (s)	Training Set	Testing Set
Logistic Regression	86.50	86.96	86.39	94.06	0.00	Positive Samples: 925 Negative Samples: 1075	Positive Samples: 930 Negative Samples: 1070
Random Forest	84.45	84.85	84.32	92.60	0.09		
SVM	87.55	87.88	87.47	94.07	0.41		
K-Nearest Neighbors	58.25	73.23	47.60	61.11	0.16		
Naive Bayes	77.50	77.66	77.45	74.84	0.00		
Gradient Boosting	77.25	80.91	76.15	87.68	0.00		
Decision Tree	75.85	75.85	75.83	77.32	0.00		
AdaBoost	71.65	76.65	69.53	75.34	0.03		

4.7 Effect of Performance with Selective ML Classifiers

- **For 2-class problem:** The performance of different machine learning models for sentiment analysis using Dataset 1 ($\mathcal{Z}_1$) is evaluated in Table 7 using accuracy, F1 score, precision, and recall. Logistic Regression achieves an accuracy of 94.49%, an F1 score of 94.67, a precision of 94.8, and a recall of 98.62. Support Vector Machine (SVM) attains an accuracy of 94.07%, an F1 score of 95.07, a precision of 94.35, and a recall of 98.57. Random Forest demonstrates an accuracy of 94.61%, an F1 score of 94.9, a precision of 94.91, and a recall of 98.67. These results indicate that all three models perform competitively, with minor variations in evaluation metrics. The corresponding performance measures for these classifiers are summarized with a visual representation of the performance is depicted in Fig. 7.
- **For 3-class problem:** The performance of different machine learning models for sentiment analysis using Dataset 2 ($\mathcal{Z}_2$) is assessed based on accuracy, F1 score, precision, and recall. Logistic Regression achieves an accuracy of 73.11%, an F1 score of 72.86, a precision of 72.76, and a recall of 84.38. Support Vector Machine (SVM) attains an accuracy of 76.09%, an F1 score of 75.29, a precision of 75.93, and a recall of 88.31. Random Forest demonstrates an accuracy of 77.69%, an F1 score of 77.31, a precision of 77.35, and a recall of 88.35. These results indicate that Random Forest outperforms the other models in terms of accuracy and F1 score, while SVM also performs well, with slightly lower accuracy but comparable recall. The corresponding performance measures for these classifiers are summarized in Table 8, while a visual representation of the performance is depicted in Fig. 8.

Table 7: Performance comparison of different classifiers applied for Dataset 1 ($\mathcal{Z}_1$).

Using f_{TFIDF}				
Classifier	**Accuracy**	**F1 Score**	**Precision**	**Recall**
LR	94.49	94.67	94.81	98.62
SVM	94.07	95.07	94.35	98.57
RF	94.61	94.90	94.91	98.67
Using f_{BoW}				
Classifier	**Accuracy**	**F1 Score**	**Precision**	**Recall**
LR	95.21	95.13	95.61	98.62
SVM	95.22	95.45	95.31	98.57
RF	95.35	95.67	95.25	98.67



(**a**) Using f_{TFIDF} (**b**) Using f_{BoW}

Figure 7: Performance comparison visualization due to different classifier models applied for Dataset 1 ($\mathcal{Z}_1$).

Table 8: Performance comparison of different classifiers applied for Dataset 2 ($\mathcal{Z}_2$).

Using f_{TFIDF}				
Classifier	**Accuracy**	**F1 Score**	**Precision**	**Recall**
LR	73.11	72.86	72.76	84.38
SVM	76.09	75.29	75.93	88.31
RF	77.69	77.31	77.35	88.35
Using f_{BoW}				
Classifier	**Accuracy**	**F1 Score**	**Precision**	**Recall**
LR	75.45	74.58	80.42	89.95
SVM	75.85	74.92	80.75	90.23
RF	76.22	75.36	81.19	90.78

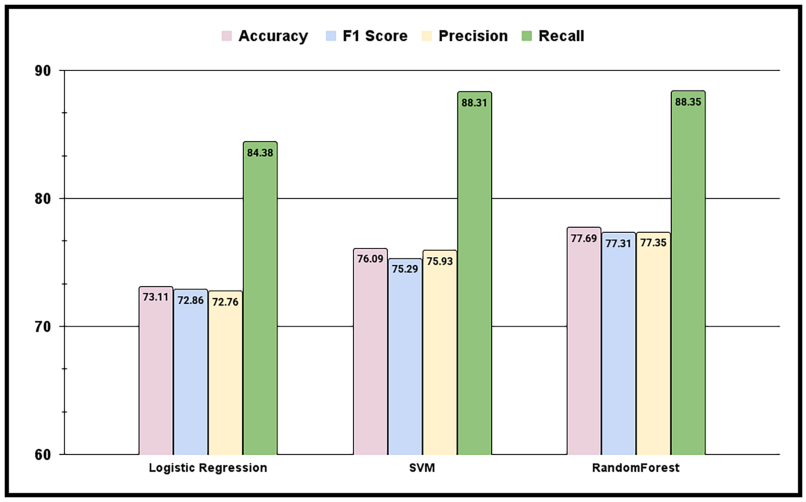


Figure 8: Performance comparison visualization due to different classifier models applied for Dataset 2 ($\mathcal{Z}_2$).

The confusion matrix for Dataset $\mathcal{Z}_2$, using the Random Forest Classifier, is shown in Table 9. This is the best-performing classifier for sentiment analysis in this experiment is the Random Forest Classifier.

Table 9: Confusion matrix for 3-class sentiment analysis.

	Predicted labels		
Actual	**Positive**	**Negative**	**Neutral**
Positive	2826	116	1349
Negative	190	2015	1685
Neutral	668	1064	611

The results presented in Table 10 highlight the impact of ensemble techniques on the performance of the proposed 3-class sentiment analysis system using two distinct feature representation schemes: f_{TFIDF}

and f_{BoW}. Random Forest (RF) classifiers trained independently on f_{BoW} and f_{TFIDF} achieved accuracy rates of 76.22% and 74.44%, respectively. We also employ sum, max, product, and average score-level fusion rules due to their low computational cost, widespread use, and suitability for real-time server-side sentiment analysis [13]. These methods require no additional trainable parameters, aligning with the objective of a lightweight and deployable client–server architecture. While these individual models demonstrated reasonable performance, combining their outputs via various fusion strategies further improved performance. The fusion rules capture complementary aggregation behaviors: sum and average smooth model variations, max selects the most confident prediction, and product reinforces inter-model agreement [37]. For comparison, single Random Forest classifiers trained on TF–IDF and BoW features are used as baselines. As shown in Table 10, all fusion methods consistently outperform the single-model baselines in the 3-class tasks. Here, it is seen that the score-Level fusion methods consistently enhanced model performance by utilizing complementary information captured by both feature extraction techniques. Among the fusion strategies, the Product Rule achieved the highest performance across all metrics, with 76.58% accuracy, 75.72% F1 score, 83.61% precision, and 76.58% recall. This superior performance indicates that combining model scores multiplicatively effectively emphasizes stronger predictions while suppressing weaker ones. The Sum Rule, Average Rule, and Max Rule fusion methods also demonstrated comparable performance, each achieving 75.58% accuracy, highlighting their robustness in combining the complementary features extracted by f_{TFIDF} and f_{BoW}. The Decision-Level Fusion method yielded slightly lower accuracy of 75.03%, suggesting that score-level fusion techniques are generally more effective for combining post-classification outputs. Overall, the results demonstrate that ensemble methods, particularly the Product Rule, offer a promising approach to improving sentiment analysis performance in multi-class scenarios by integrating diverse feature representation strategies without added architectural complexity.

- **For 13-class problem:** For thirteen class sentiment analysis system ('empty', 'sadness', 'enthusiasm', 'neutral', 'worry', 'surprise', 'love', 'fun', 'hate', 'happiness', 'boredom', 'relief', 'anger'), initially area under the curve (AUC) has been performed for Dataset 3 ($\mathcal{Z}_3$) before data augmentation with class imbalance at Fig. 9, and after data augmentation with class balance in Fig. 10. Fig. 9 shows that the ROC curves compare the classification performance of Logistic Regression, SVM, and Random Forest across different emotions. The SVM model demonstrates the best performance overall, with relatively higher AUC values (e.g., empty = 0.73, sadness = 0.61), indicating better separability. Logistic Regression and Random Forest show moderate to poor performance, with some emotions (e.g., love, worry, hate) exhibiting very low AUC values, suggesting difficulty in distinguishing these classes. These results highlight the importance of model selection and the need for better feature representations to improve classification.

Table 10: Effect of ensemble of trained models of f_{TFIDF} and f_{BoW} using different post-classification fusion methods on the performance of the proposed 3-class sentiment analysis system.

Model	Accuracy (%)	F1 Score (%)	Precision (%)	Recall (%)
RF (f_{BoW})	76.22	75.36	81.19	76.22
RF (f_{TFIDF})	74.44	73.48	80.97	74.44
Decision-Level Fusion	75.03	73.62	81.12	75.03
Score-Level Fusion (Sum Rule)	75.58	74.73	81.60	75.58
Score-Level Fusion (Product Rule)	76.58	75.72	83.61	76.58
Score-Level Fusion (Max Rule)	75.58	74.71	81.55	75.58
Score-Level Fusion (Min Rule)	75.58	74.73	81.55	75.58
Score-Level Fusion (Average Rule)	75.58	74.73	81.60	75.58

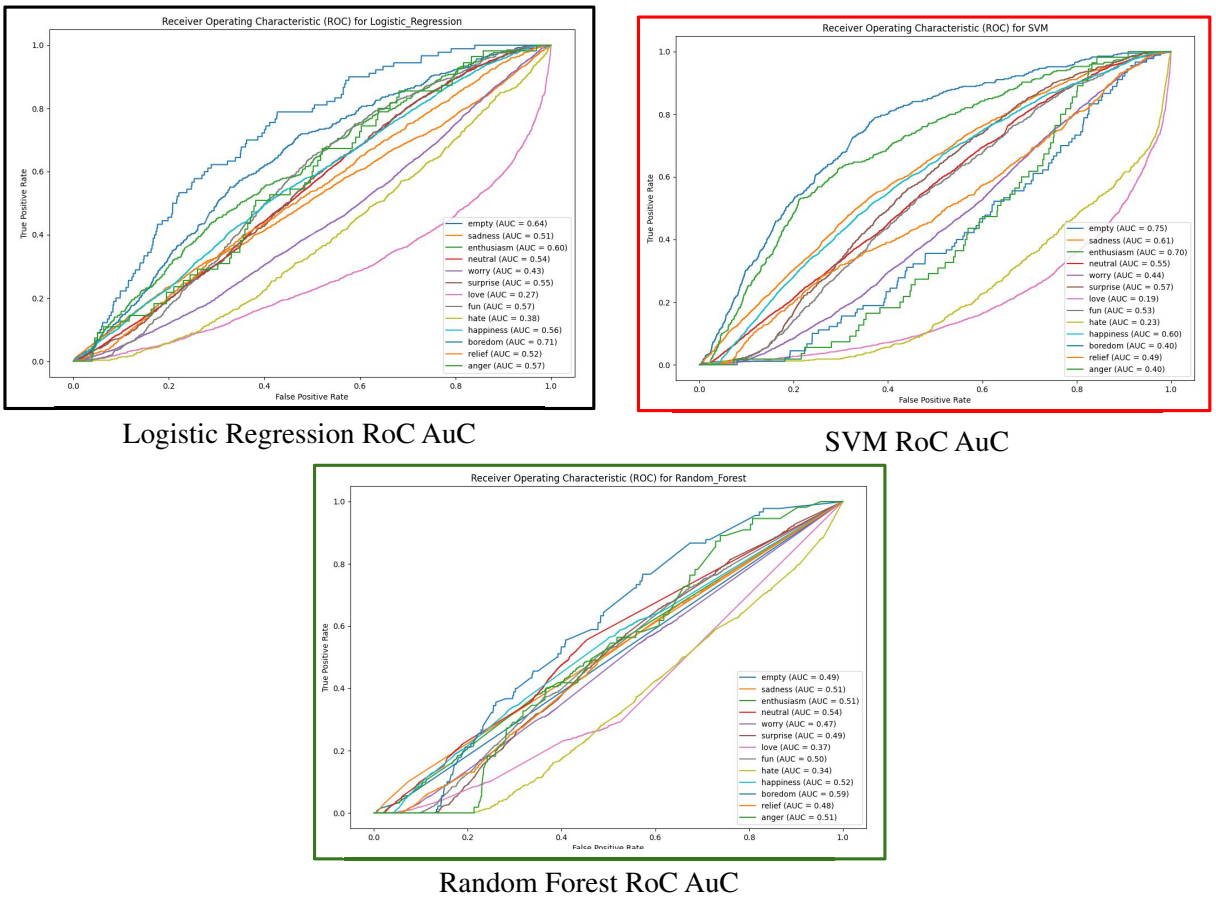


Figure 9: Multiple comparison of AUC vs. Feature vector size for logistic regression, SVM, random forest model for Dataset 3 ($\mathcal{Z}_3$) before data augmentation with class imbalance.

Fig. 10 shows that the ROC curves illustrate the comparative performance of Logistic Regression, SVM, and Random Forest classifiers across various emotion classes. SVM achieves the highest AUC values for most emotions (e.g., empty = 0.84, sadness = 0.63), indicating better classification capability. Logistic Regression performs moderately well, but some emotions (e.g., love, worry, hate) show poor separability. Random Forest demonstrates the lowest AUC values overall, signifying weaker classification performance. These results suggest that SVM is the most effective model in this setup, while Random Forest struggles, likely due to the complexity of feature interactions.

Hence, comparing the pre-augmentation and post-augmentation ROC curves shows that data augmentation improves classification performance across all models. Before augmentation, the AUC values for most classes were lower, indicating weaker separability of emotions. Models like Random Forest struggled significantly, and even SVM showed suboptimal performance in certain classes. After augmentation, there is a notable increase in AUC scores, particularly for SVM and Logistic Regression, suggesting that the models benefit from the increased variability and robustness in training data. For instance, emotions like empty and sadness exhibit higher AUC values in all classifiers post-augmentation. The improvement is most significant in SVM, reinforcing its strength in high-dimensional feature spaces. Random Forest also improves, but it still lags behind the other models. Overall, data augmentation enhances generalization, leading to better class separability and improved classification performance across all models. The balanced 13-sentiment analysis dataset has now been employed for further experiments.

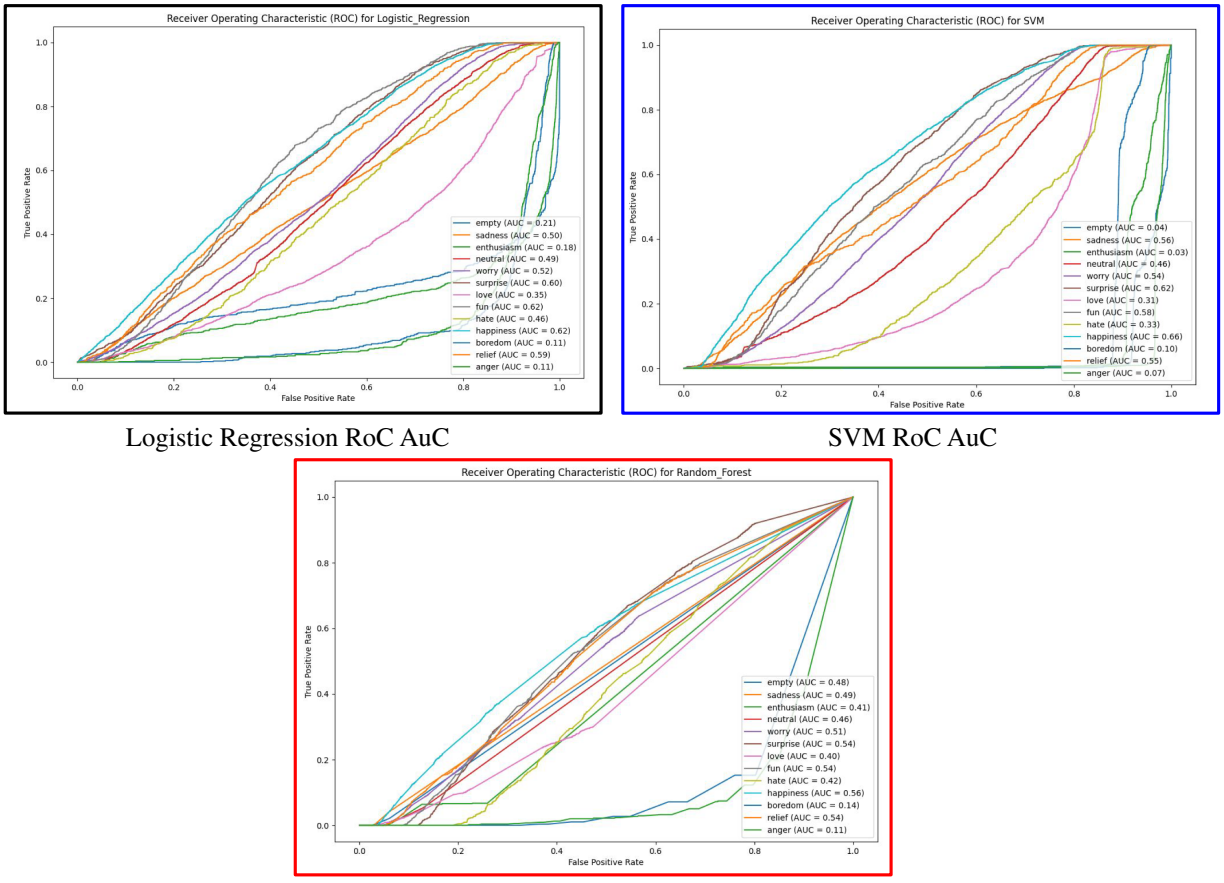


Figure 10: Multiple comparison of AUC vs. Feature vector size for logistic regression, SVM, random forest model for Dataset 3 ($\mathcal{Z}_3$) after data augmentation with class balance.

The performance of different machine learning models for sentiment analysis is evaluated on Dataset 3 ($\mathcal{Z}_3$) using accuracy, F1 score, precision, and recall in Table 11. The performance metrics presented in this Table illustrate the effectiveness of different classifiers using two distinct feature extraction methods: TF-IDF (f_{TFIDF}) and Bag-of-Words (f_{BoW}). For TF-IDF, the Random Forest (RF) classifier achieves the highest accuracy (95.83%) and F1 Score (95.85%), significantly outperforming Logistic Regression (LR) and Support Vector Machine (SVM), which exhibit lower performance across all metrics. Conversely, using the BoW feature extraction method, RF again demonstrates superior performance with 94.91% accuracy and 94.94% F1 Score, followed by SVM with 85.42% accuracy and LR with 80.17% accuracy. The improved results for RF in both feature extraction approaches highlight its robustness and reliability in text classification tasks, while SVM performs moderately well with BoW features but lags behind with TF-IDF. LR consistently underperforms compared to the other models, demonstrating limited effectiveness across both feature extraction techniques. These comparisons among Logistic Regression, Support Vector Machine (SVM), and Random Forest Classifier for sentiment analysis across thirteen classes—empty, sadness, enthusiasm, neutral, worry, surprise, love, fun, hate, happiness, boredom, relief, and anger—is presented in Table 11. The corresponding performance measures are visually depicted in Fig. 11.

Table 11: Performance comparison of different classifiers applied for Dataset 3 ($\mathcal{Z}_3$).

Using f_{TFIDF}				
Classifier	**Accuracy**	**F1 Score**	**Precision**	**Recall**
LR	58.77	56.45	61.82	41.32
SVM	78.02	76.32	81.67	37.27
RF	95.83	95.85	95.97	99.69
Using f_{BoW}				
Model	**Accuracy**	**F1 Score**	**Precision**	**ROC AUC**
SVM	85.42	85.45	85.82	89.63
LR	80.17	80.20	80.62	84.66
RF	94.91	94.94	95.13	99.59

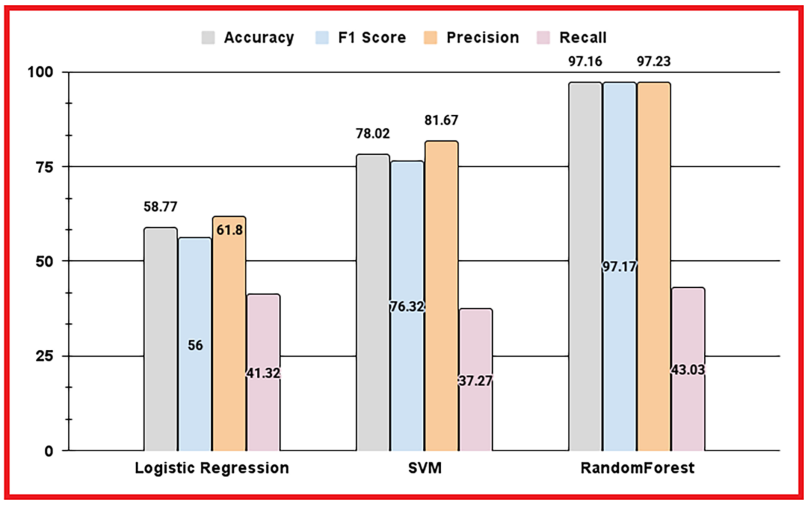


Figure 11: Performance comparison visualization due to different classifier models applied for Dataset 3 ($\mathcal{Z}_3$).

The thirteen sentiment classes are grouped into three broader sentiment categories: **Positive Emotions:** Enthusiasm, Love, Fun, Happiness, Relief; **Negative Emotions:** Sadness, Worry, Hate, Anger, Boredom; **Neutral Emotions:** Empty, Neutral, Surprise. Here, the derived model for Random Forest Classifier for sentiment analysis across thirteen classes have been employed to build the confusion matrix which is shown in Table 12. The computed performance metrics from the given confusion matrix are as follows: the average precision is 95.25%, the average recall is 94.63%, the average F1-score is 94.92%, and the overall accuracy is 96.12%. These metrics demonstrate a strong classification performance across all sentiment categories. The high precision value indicates that the model makes few false positive errors, while the high recall suggests it correctly identifies most actual instances of each sentiment. The F1-score, which balances precision and recall, remains consistently high, reinforcing the model's robustness. Moreover, the overall accuracy of 96.12% highlights that the classifier is correctly predicting sentiment labels for the majority of instances, confirming its effectiveness in sentiment analysis.

Table 13 presents a comprehensive performance comparison of the proposed 13-class sentiment analysis system using different feature representations and post-classification fusion strategies. The evaluation

includes Accuracy, Macro-F1, Weighted-F1 (W-F1), Weighted Precision, and Weighted Recall, reported in percentage terms to enable consistent comparison across models. The models considered include Random Forest classifiers trained using two distinct feature representation techniques, namely f_{BoW} and f_{TFIDF}, along with multiple ensemble techniques that combine these models using different post-classification fusion strategies. The evaluation metrics include Accuracy, F1 Score, Precision, and Recall, expressed as percentages to facilitate performance comparison.

Table 12: Confusion matrix for 13-class sentiment analysis.

Actual	Predicted labels												
	Empty	Sadness	Enthusiam	Neutral	Worry	Suprise	Love	Fun	Hate	Happiness	Boredom	Relief	Anger
Empty	910	1	0	3	0	0	0	0	0	0	0	0	0
Sadness	0	2551	2	38	19	2	2	0	4	1	0	0	0
Enthusiam	0	0	877	3	0	0	0	0	0	0	0	0	0
Neutral	0	6	0	4240	12	6	33	1	2	16	0	3	0
Worry	0	13	0	85	4105	5	15	2	1	4	0	0	0
Suprise	0	3	0	31	5	1036	14	0	0	5	0	0	0
Love	0	0	0	31	4	0	1864	7	0	15	0	0	0
Fun	0	1	0	9	2	0	2	864	5	4	0	1	0
Hate	0	3	0	5	2	0	0	0	645	7	0	0	0
Happiness	0	0	0	73	4	1	46	2	0	2472	6	1	0
Boredom	0	0	0	0	0	0	0	0	0	0	585	5	0
Relief	0	1	0	27	3	0	5	0	0	6	0	713	8
Anger	0	0	0	0	0	0	0	0	0	0	0	0	555

Table 13: Effect of ensemble of trained models of f_{TFIDF} and f_{BoW} using different post-classification fusion methods on the performance of the proposed 13-class sentiment analysis system.

Model	Acc.	Macro-F1	W-F1	Prec. (W)	Rec. (W)
RF (f_{BoW})	94.91	94.59	94.94	95.13	94.91
RF (f_{TFIDF})	95.83	95.47	95.85	95.97	95.83
Decision-Level Fusion	94.85	94.91	95.14	95.67	94.85
Score-Level Fusion (Sum)	95.86	95.83	95.88	96.01	95.86
Score-Level Fusion (Product)	95.97	95.87	95.91	95.95	95.97
Score-Level Fusion (Max)	95.84	94.29	95.86	95.98	95.84
Score-Level Fusion (Min)	95.72	94.97	95.75	95.89	95.72
Score-Level Fusion (Average)	95.86	95.11	95.88	96.01	95.86

From the results presented in Table 13, it has been observed that the Random Forest model trained with f_{TFIDF} achieves the highest accuracy (95.83%), demonstrating its superior ability to handle complex text data through improved feature representation. The Decision-Level Fusion method also demonstrates strong performance, achieving an F1 Score of 95.14%, indicating robust precision-recall balance. Among the score-level fusion techniques, the Product Rule achieves the best accuracy (95.97%) and maintains competitive scores across other metrics, making it the most effective fusion strategy in this evaluation. Other score-level fusion methods, including the Sum and Average rules, also deliver competitive results, achieving accuracies above 95.8%. However, their Macro-F1 scores are slightly lower than those of the Product Rule, suggesting comparatively weaker performance on minority classes. The Max and Min rules show marginally reduced Macro-F1 values, particularly the Max rule (94.29%), indicating reduced stability in balanced class performance despite maintaining high overall accuracy. We have also evaluated the model performance based on per-class F1 score for product fusion, which is illustrated in Fig. 12.

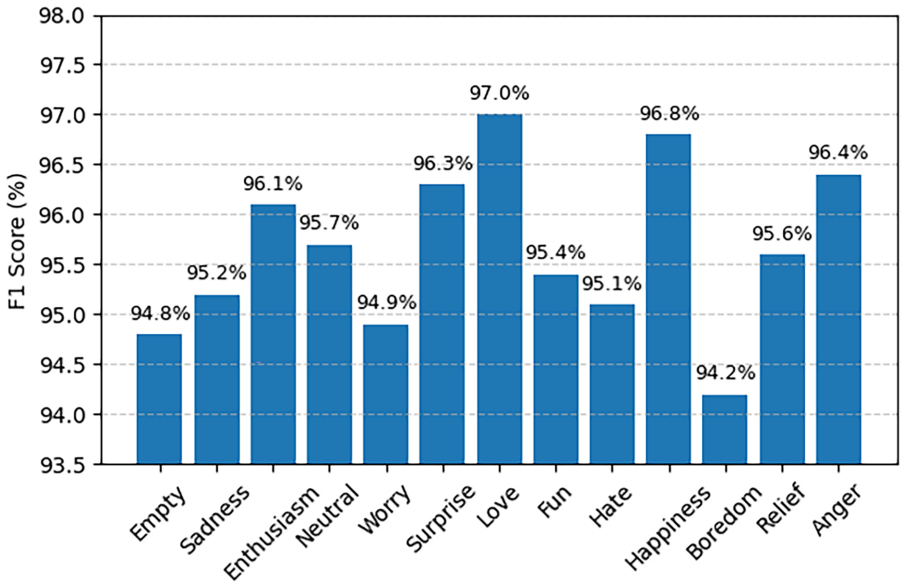


Figure 12: Per-class F1 score for 13-class emotion for product fusion.

The results emphasize that fusion strategies improve performance by utilizing the strengths of both f_{BoW} and f_{TFIDF} feature representations. Notably, the Product Rule surpasses individual models in overall accuracy, demonstrating the benefits of combining multiple decision outputs. Consequently, ensemble methods are effective for improving the predictive performance of sentiment analysis models, especially in multi-class classification tasks. These results highlight the importance of incorporating diverse fusion strategies to achieve optimal model robustness and accuracy in text classification systems.

4.7.1 Ablation Analysis of Fusion Strategies

To systematically evaluate the contribution of the fusion mechanism, we conducted a structured ablation study comparing individual feature-stream models and multiple fusion strategies. The objective is to quantify the performance gain introduced by score-level fusion relative to standalone classifiers. Table 14 presents the structured ablation analysis comparing single-stream baselines and multiple fusion strategies based on accuracy. The results clearly indicate that score-level fusion consistently outperforms standalone models across all classification settings. Among the evaluated strategies, the Product rule achieves the highest performance. Compared to the best individual model over multiple independent run, the observed performance gain (Δ) in Table 15 confirms that the dual feature streams provide complementary predictive information rather than redundant signals.

4.7.2 Analytical Justification of Product Rule Fusion

Among the evaluated score-level fusion strategies, the Product rule consistently achieves the highest performance across multiple classification settings. This behavior can be theoretically explained through its emphasis on inter-model agreement. Let $P_1(c)$ and $P_2(c)$ denote the posterior class probabilities predicted by the classifiers trained on f_{BoW} and f_{TFIDF}, respectively. The Product fusion strategy computes the combined confidence as: $P_{fusion}(c) = P_1(c) \times P_2(c)$. This multiplicative formulation has two important properties:

1. **Amplification of consensus:** When both models assign high confidence to the same class, the product increases disproportionately, strengthening agreement-based predictions.

2. **Suppression of disagreement:** If one model assigns a low confidence to a class, the resulting product remains low, effectively penalizing inconsistent predictions.

Table 14: Ablation study: comparison of single models and fusion strategies. Results are reported as Mean ± Standard deviation over multiple independent runs.

Method	2-Class	3-Class	13-Class
RF (BoW)	95.35 ± 0.12	76.22 ± 0.18	94.91 ± 0.15
RF (TF-IDF)	94.61 ± 0.14	74.44 ± 0.21	95.83 ± 0.10
Decision-Level Fusion	94.15 ± 0.20	75.03 ± 0.19	94.85 ± 0.17
Score Fusion (Sum)	95.29 ± 0.11	75.58 ± 0.16	95.86 ± 0.09
Score Fusion (Product)	95.67 ± 0.10	76.58 ± 0.15	95.97 ± 0.07
Score Fusion (Average)	95.35 ± 0.11	75.58 ± 0.16	95.86 ± 0.09
Score Fusion (Max)	94.51 ± 0.18	75.58 ± 0.17	95.84 ± 0.11
Score Fusion (Min)	94.49 ± 0.19	75.58 ± 0.17	95.72 ± 0.12

Table 15: Ablation study with performance gain over best single model (Accuracy). All values are reported as Mean ± Standard Deviation over multiple independent runs.

Method	2C	3C	13C
Best Single Model	95.35 ± 0.12	76.22 ± 0.18	95.83 ± 0.09
Score Fusion (Product)	95.67 ± 0.10	76.58 ± 0.15	95.97 ± 0.07
Δ (Mean Improvement)	+0.32	+0.36	+0.14

Compared to additive strategies such as the Sum or Average rules, the Product rule is more selective because it does not allow a single high-confidence prediction to dominate the final decision when the other model is uncertain. Instead, it favors classes that receive consistently strong support from both feature streams. This consensus-driven behavior is particularly beneficial in multi-class emotion recognition tasks, where class boundaries are often overlapping and misclassification risk increases due to semantic similarity. By reinforcing agreement and attenuating conflicting signals, Product fusion improves discriminative robustness without introducing additional trainable parameters.

4.7.3 Quantitative Validation of LLM-Based Data Augmentation

To rigorously assess the contribution of LLM-based synthetic data augmentation, we conducted a controlled ablation study isolating its effect on model performance. The comparison includes model performance trained on (i) the original human-annotated dataset and (ii) the augmented dataset incorporating synthetic samples. All this evaluation was done for multiple run for consistency. Importantly, synthetic data were used exclusively during training, while test sets contained only human-annotated samples to prevent data leakage. Table 16 presents the overall performance comparison before and after augmentation. The results show a consistent improvement across all evaluation metrics.

Specifically, the augmented model demonstrates an increase in Macro-F1, indicating improved balanced multi-class performance. It also shows an improvement in Macro-AUC, reflecting enhanced class separability. The gains in Weighted-F1 and Accuracy, confirming that augmentation does not adversely affect majority classes. The observed Δ improvements confirm that the performance gain is attributable to training-time augmentation rather than evaluation bias.

Table 16: Impact of LLM-based data augmentation on overall performance.

Metric	Before Aug.	After Aug.	Δ Improvement
Macro-F1	0.9469	0.9587	+0.0118
Weighted-F1	0.9579	0.9591	+0.0012
AUC (Macro)	0.7179	0.7439	+0.026
Accuracy	93.89	95.97	+2.08

4.8 Sustainable Model Deployment to Server

In this work, the obtained 3-class and 13-class sentiment analysis model deployment has been performed in the Salesforce-licensed cloud service. A Salesforce-licensed cloud service has been acquired exclusively for academic research to deploy the sustainable sentiment analysis model for real-time processing. While Salesforce primarily offers cloud solutions for business and commercialization, it is utilized here solely for research deployment. The sentiment analysis application features a user-friendly graphical interface where users can input text for analysis. Upon clicking the "Analyze" button, the entered text is processed and forwarded to the sentiment analysis model developed using the previously described methodology. The model evaluates the sentiment and generates a classification result displayed within the GUI, providing users with immediate feedback.

For deploying the models to the server, the trained models f_{TFIDF} and f_{BoW} are further examined to determine the optimal dimensionality of the feature vectors. A vector quantizer is applied to both models to identify the minimum feature vector dimension that remains effective for sentiment classification. If a lower-dimensional feature vector proves sufficient, that model will be deployed to the busy server to reduce both space requirements and computational complexity, ensuring efficient resource utilization. The Table 17 presents the performance metrics for different feature vector sizes of f_{BoW} applied to Dataset 3 ($\mathcal{Z}_3$), demonstrating the impact of dimensionality reduction on classification effectiveness. As the feature vector size decreases from **2000 to 500**, there is a gradual decline in accuracy, F1 score, precision, and recall. This reflects the inherent trade-off between computational efficiency and model performance. While higher-dimensional feature representations capture more intricate details, lower-dimensional ones are more efficient for real-world deployment, ensuring a balance between accuracy and resource constraints.

Table 17: Performance metrics for different feature vector sizes of f_{BoW} applied on Dataset 3 ($\mathcal{Z}_3$).

Feature Vector Size	Accuracy (%)	F1 Score (%)	Precision (%)	Recall (%)
2000	94.91	94.94	95.13	94.91
1500	93.82	93.79	94.07	93.80
1000	92.68	92.76	92.98	92.70
700	91.59	91.63	91.78	91.60
600	90.44	90.50	90.66	90.42
500	89.29	89.35	89.51	89.27

The performance metrics for various feature vector sizes of f_{TFIDF} are shown in Table 18, which is characterized by a distinct trend in accuracy, F1 score, precision, and recall as the feature dimension increases. Achieving 95.83% accuracy at 2000 dimensions yields the best performance, with performance gradually declining as feature size decreases. This decrease is anticipated because lower-dimensional representations

lose information, resulting in a slight reduction in classification performance. Nevertheless, the model performs reasonably well at 500 dimensions, with an accuracy of 89.25%, suggesting that the TF-IDF-based model remains reliable even with reduced computational complexity.

Table 18: Performance metrics for different feature vector sizes of f_{TFIDF} applied on Dataset 3 ($\mathcal{Z}_3$).

Feature Vector Size	Accuracy (%)	F1 Score (%)	Precision (%)	Recall (%)
2000	95.83	95.85	95.97	95.83
1500	94.68	94.70	94.82	94.66
1000	93.27	93.25	93.37	93.21
700	91.98	91.95	92.07	91.91
600	90.68	90.65	90.77	90.61
500	89.25	89.24	89.36	89.20

For model deployment to the server, selecting an optimal feature vector size requires balancing computational efficiency and model performance. The results indicate that both f_{BoW} and f_{TFIDF} achieve the highest accuracy at 2000 dimensions, but as the feature size decreases, a slight performance drop is observed. While reducing the feature dimension to 500 significantly reduces computational costs, it still maintains a reasonable accuracy of 89.29% for f_{BoW} and 89.25% for f_{TFIDF}. Given this trend, a feature vector size of 1000 appears to be a suitable choice for deployment, as it offers a balance between accuracy (92.68% for f_{BoW} and 93.27% for f_{TFIDF}) and reduced memory footprint, ensuring efficient real-time processing without a substantial loss in sentiment classification effectiveness.

The sustainable sentiment analysis model has been deployed on the **Ultimate Coder** platform (`sa.ultimatecoder.in`), which is currently hosted by Salesforce Consulting Company. This deployment includes both the **3-class** and **13-class** sentiment analysis models. Fig. 13 presents real-time testing snapshots of the **Graphical User Interface (GUI)**, illustrating an example of **3-class sentiment analysis** performed using the deployed model. Similarly, Fig. 14 showcases real-time sample text evaluations demonstrating the performance of the **13-class sentiment analysis** model.

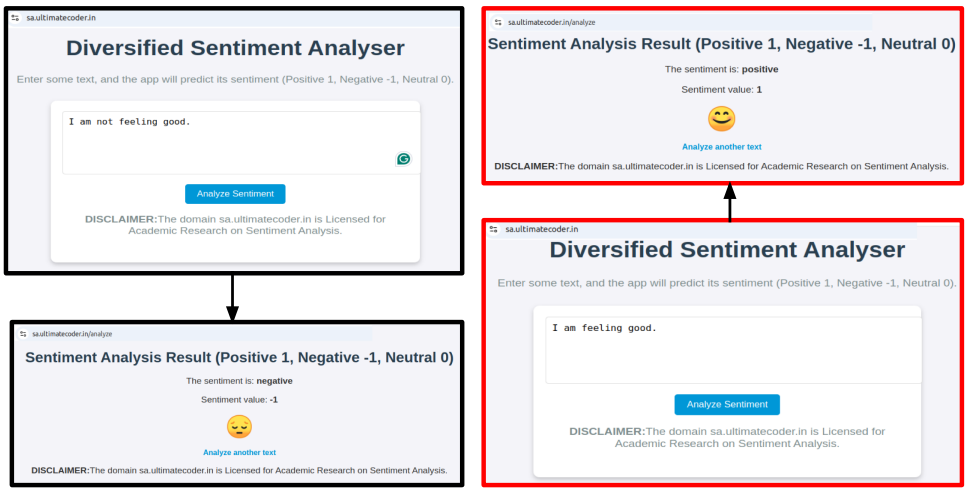


Figure 13: Graphical User Interface (GUI) implementation with an example showcasing 3-class sentiment analysis tested on the applied model.

Figure 14: Graphical User Interface (GUI) implementation with an example showcasing 13-class sentiment analysis tested on the applied model.

Table 19 presents the model complexity analysis for deploying a sentiment classification model with a 1000-dimensional feature vector, considering Bag-of-Words (BoW) and Term Frequency-Inverse Document Frequency (TF-IDF) representations. The total parameters represent the number of trainable and non-trainable elements in the model, influencing its storage and computational efficiency. The memory footprint (in megabytes, MB) indicates the storage required to load the model into memory, with both BoW (180 MB) and TF-IDF (190 MB) being relatively lightweight for deployment. The computational cost, measured in floating point operations per second (FLOPs), quantifies the number of arithmetic operations needed for inference, with BoW requiring approximately 4.2 GFLOPs (giga floating point operations) and TF-IDF needing 4.5 GFLOPs, making both computationally feasible. Given these factors, a 1000-dimensional feature vector offers a balance between performance and efficiency, ensuring a suitable trade-off for server deployment, minimizing storage and computational overhead while maintaining high classification accuracy.

Table 19: Model complexity analysis for server deployment with a 1000-dimensional feature vector.

Feature Model	Total Parameters	Memory Footprint (MB)	Computational Cost (FLOPs)	Suitability for Deployment
f_BoW	Moderate	180 MB	4.2 GFLOPs	Suitable trade-off
f_TFIDF	Moderate	190 MB	4.5 GFLOPs	Suitable trade-off

5 Results and Discussion

The system classifies individual text inputs and reports the corresponding results. To improve performance, the sentiment analysis is extended to a second level, following a two-tier approach. The first level categorizes sentiment into three broad groups, forming the basis for a more detailed classification into thirteen sentiment classes. The subsequent section provides an in-depth discussion of the text-based sentiment analysis. To study the performance of our proposed model with the existing client-server model, we have fixed metrics like inference location, inference time, end-to-end latency, network overhead, latency variability and sentiment class. The existing models are categorized into three types: client-side (Edge) model, traditional cloud server model and hybrid (split-inference) model. Each of the models are generalised with empirical data for each evaluation metrics. Table 20 represents the analysis of each model.

Table 20: Performance comparison of existing client–server architectures.

Metric	Client-Side (Edge)	Traditional Cloud Server	Hybrid (Split-Inference)
Inference Location	User device (phone/PC)	Remote data center	Edge (pre-processing) + cloud
Inference Time	~10–50 ms (device dependent)	~5–20 ms (high-performance GPU)	~15–30 ms (distributed inference)
End-to-End Latency	Low (<60 ms)	High (>150–500 ms)	Moderate (80–120 ms)
Network Overhead	Negligible (local processing)	High (uploads raw data/video)	Low (transmits only features/vectors)
Latency Variability	Minimal (stable local environment)	High (network congestion risk)	Moderate (synchronization overhead)
Sentiment Classes	3–5 basic (e.g., happy, sad, angry)	7–20+ complex (e.g., joy, awe, irony)	5–10 nuanced (scalable)

The in-depth study of existing models suggests a need for substantial computational resources to host the model, due to latency and network overhead, which diverges from our research aim. To maintain the integrity and novelty of our research we opt for light weight client-server model. The resource utilization by the server when the random forest model is hosted and when a sentiment analysis prediction is made by the model are illustrated in Table 21. From the server physical memory usage of 2 GB RAM, 5̃00 MB and 1̃350 MB are utilized for running flask and other processes, the remaining 8̃50 MB is used by the proposed model for prediction process.

Table 21: System resource usage: hosting vs prediction phases.

Description	Phase	Usage	Limit	Fault
CPU Usage	Hosting	2.4%	200%	0
	Prediction	0	200%	0
I/O Usage	Hosting	162.4 KB/s	48.83 MB/s	0
	Prediction	0	48.83 MB/s	0
IOPS	Hosting	7	2048	0
	Prediction	0	2048	0
NPROC	Hosting	0	100	0
	Prediction	6	100	0
Entry Processes	Hosting	0	30	0
	Prediction	2	30	0
Physical Memory Usage	Hosting	500.15 MB	2 GB	0
	Prediction	1.35 GB	2 GB	0

We used the traditional machine learning approach to keep the computational cost for prediction as minimum as possible. To compare the performance of our model, we chose VADER (Valence Aware Dictionary and sEntiment Reasoner), one of the most poplular lexicon-based sentiment analysis model for informal text sentiment analysis. VADER is a three-class sentiment analyzer, while our model can support emotions upto thirteen classes. So, to evaluate the performance, we limit our model to three classes, i.e, positive, negative and neutral and test the performance based on metrics, that are used to evaluate client-server architecture model, which is illustrated in Table 22.

Table 22: Comparison of VADER and proposed 13-class sentiment model with respect to different metrics.

Metric	VADER	Proposed 13-Class Model
Inference Location	Client-side	Server-side
Inference Time	~0.2–1.0 ms	~10–25 ms
End-to-End Latency	~0.3–1.5 ms	~30–70 ms
Network Overhead	None	Present
Latency Variability	High	Low
Sentiment Classes	3	13

A comparison of Random Forest and VADER in terms of system memory usage and processing time is illustrated in Fig. 15.

Random Forest exhibits significantly higher memory usage, while VADER achieves lower inference time with minimal resource overhead and latency. We futhermore, hosted the model in the given environment and performed a deep analysis on these models, which is shown in Table 23.

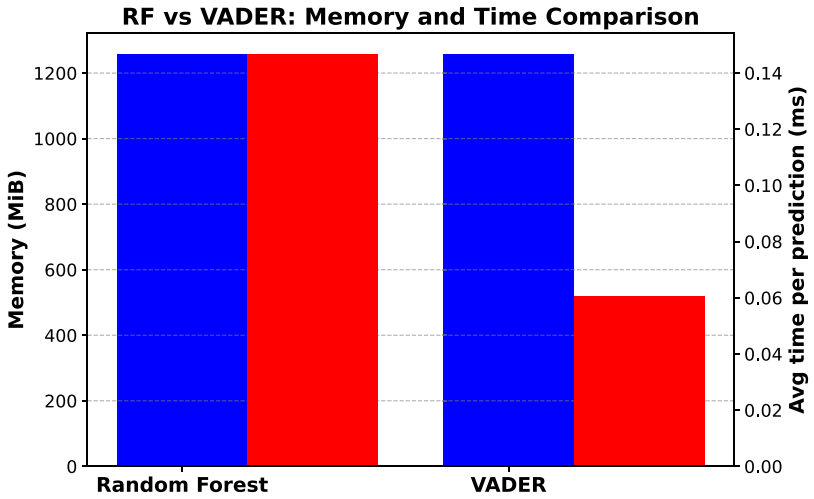


Figure 15: Comparison of Random Forest and VADER in terms of memory consumption and average prediction time.

Table 23: Client–server–level comparison of VADER and proposed 13-class sentiment model.

Parameter	VADER Sentiment Analyzer	Proposed Server-Based Sentiment Model
Inference Location	Client-side	Server-side
Architecture Type	Standalone library	REST-based client–server architecture
Sentiment Granularity	3-class sentiment with compound score	13-class fine-grained sentiment classification
Supported Output Labels	Positive, Neutral, Negative	13 sentiment/emotion labels
Response Format	Local dictionary output {`pos: 0.62, neg: 0.08, neu: 0.30`}	Structured JSON response
Output Metadata	Not available	Confidence score, inference latency, model information
Multi-Class Support	Limited	Native multi-class (13-class) support
Inference Latency	Very low (rule-based processing)	Optimized and predictable server-side latency
Network Overhead	None	Present due to API communication
Scalability	Limited (dependent on client hardware)	High (supports multiple concurrent client requests)

The models are additionally compared based on their response format and interface design, where the proposed system employs a structured JSON-based API response enabling richer sentiment representation,

confidence reporting, and latency measurement, unlike VADER's local dictionary output. Table 24 illustrates the JSON format output.

Table 24: Comparison of output responses for VADER and proposed sentiment models.

Category	Model	Output Response Details
3-Class Sentiment Output	VADER	• Input Text: *"The product quality is excellent except the color"* • neg: 0.05, neu: 0.25, pos: 0.70 • compound: 0.86
	Proposed Model	• Request ID: REQ_1042 • Input Text: *"The product quality is excellent except the color"* • Predicted Sentiment: Positive • Confidence: 0.94 • Model: TF-IDF + RF (fusion) • Latency: 38 ms
13-Class Sentiment Output	Proposed Model	• Request ID: REQ_1297 • Input Text: *"I feel extremely frustrated with the service"* • Predicted Label: Anger • Confidence: 0.91 • Class Scores: – happiness (0.01), sadness (0.03), anger (0.91), hate (0.08), love (0.02), fun (0.01), surprise (0.01), empty (0.01), enthusiasm (0.01), relief (0.00), neutral (0.01), worry (0.01), boredom (0.00) • Model: TF-IDF + RF (score-level fusion) • Latency: 42 ms

Unlike VADER, which is limited to coarse-grained polarity estimation, the proposed system performs fine-grained 13-class sentiment classification within a client–server architecture. The server returns structured JSON responses containing the predicted sentiment label, confidence score, class-wise probabilities, and latency metrics. Although the multi-class inference introduces additional computational overhead, centralized server-side processing ensures predictable latency while enabling richer emotional understanding and scalability.

6 Conclusions

This work presents a sustainable sentiment analysis system utilizing textual data, designed for real-time deployment within a client-server architecture. The system is structured into five key components: text preprocessing, feature extraction, sentiment classification, model optimization, and server deployment. Two feature representation techniques, Bag-of-Words (BoW) and Term Frequency-Inverse Document Frequency (TF-IDF), are evaluated for their effectiveness in sentiment classification. The models are trained and tested on three benchmark datasets corresponding to two-class, three-class, and thirteen-class sentiment categories. For model deployment to the server, selecting an optimal feature vector size requires balancing computational efficiency and classification performance. The results indicate that both f_{BoW} and f_{TFIDF}

achieve peak accuracy at 2000 dimensions, but a gradual decline is observed as the feature size decreases. While reducing the feature dimension to 500 significantly lowers computational costs, it still maintains reasonable accuracy—89.29% for f_{BoW} and 89.25% for f_{TFIDF}. Given this trade-off, a feature vector size of 1000 is identified as the optimal choice for deployment, achieving a balance between accuracy (92.68% for f_{BoW} and 93.27% for f_{TFIDF}) and reduced memory footprint, ensuring efficient real-time sentiment classification with minimal computational overhead. However the proposed framework achieves competitive accuracy with low computational overhead, it has several limitations. First, the reliance on classical feature representations (BoW and TF-IDF) limits the model's ability to capture long-range semantic dependencies and contextual nuances present in complex or figurative language. Second, the system is currently evaluated on monolingual, benchmark datasets, which may restrict its generalization to multilingual or domain-shifted real-world scenarios. Third, although feature dimensionality is optimized for efficiency, the static feature size does not adapt dynamically to evolving data distributions or sentiment drift. Future work will address these limitations by incorporating context-aware deep learning representations, extending the framework to multilingual and cross-domain sentiment analysis, and exploring adaptive and incremental learning strategies for dynamic environments. These directions aim to enhance the robustness and scalability of the proposed system while preserving its emphasis on efficient, real-time deployment.

Acknowledgement: None.

Funding Statement: The authors received no specific funding for this study.

Author Contributions: Conceptualization: Saiyed Umer, Soumalya De, and Rahil Akhtar; Methodology: Ranjeet Kumar Rout, Saiyed Umer, and Soumalya De; Software: Soumalya De, Saiyed Umer, and Rahil Akhtar; Validation: Saiyed Umer, Soumalya De, Rahil Akhtar, Ranjeet Kumar Rout, and G. G. Md. Nawaz Ali; Formal analysis: Saiyed Umer, Soumalya De, and G. G. Md. Nawaz Ali; Investigation: Saiyed Umer, and Ranjeet Kumar Rout; Resources: Rahil Akhtar, Soumalya De, and Saiyed Umer; Data curation: Rahil Akhtar, and Soumalya De; Writing the original draft preparation: Saiyed Umer, Rahil Akhtar, and Soumalya De; Writing—review and editing, Soumalya De, Ranjeet Kumar Rout, and G. G. Md. Nawaz Ali; Visualization, G. G. Md. Nawaz Ali; Supervision, Saiyed Umer, Soumalya De, Ranjeet Kumar Rout, and G. G. Md. Nawaz Ali; Project administration, Saiyed Umer, G. G. Md. Nawaz Ali. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: The authors confirm that the datasets utilized in this study are publicly available, and the corresponding links are https://huggingface.co/datasets/tasksource/crowdflower/viewer/tweet%20global%20warming, https://huggingface.co/datasets/tasksource/crowdflower/viewer/airline-sentiment, and https://huggingface.co/datasets/tasksource/crowdflower/viewer/textemotion?views%5B%5D=textemotion. These links have been provided in the relevant sections of the manuscript.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Kadu S, Joshi B. Text-based sentiment analysis using deep learning techniques. In: Deep learning for social media data analytics. Cham, Switzerland: Springer; 2022. p. 81–100.
2. Zad S, Heidari M, Jones JH, Uzuner O. A survey on concept-level sentiment analysis techniques of textual data. In: Proceedings of the IEEE World AI Internet of Things Congress (AIIoT). Piscataway, NJ, USA: IEEE; 2021. p. 285–91.
3. Lin T, Wang Y, Liu X, Qiu X. A survey of transformers in natural language processing: history, progress, and future directions. ACM Comput Surv. 2022;54(10):1–41.

4. Adagale SS, Gupta P. Comprehensive analysis of text based sentiment analysis using deep learning. In: Proceedings of the IEEE International Conference on Information Technology, Electronics and Intelligent Communication Systems (ICITEICS). Piscataway, NJ, USA: IEEE; 2024. p. 1–5.
5. Li Y, Ding H, Lin Y, Feng X, Chang L. Multi-level textual-visual alignment and fusion network for multimodal aspect-based sentiment analysis. Artif Intell Rev. 2024;57(4):78. doi:10.1007/s10462-023-10685-z.
6. Vijayaragavan P, Suresh C, Maheshwari A, Vijayalakshmi K, Narayanamoorthi R, Gono M, et al. Sustainable sentiment analysis on e-commerce platforms using a weighted parallel hybrid deep learning approach for smart cities applications. Sci Rep. 2024;14(1):26508. doi:10.1038/s41598-024-81475-y.
7. Mohammed AMK, Ali GGMN, Khairunnesa SS. GSAF: an ML-based sentiment analytics framework for understanding contemporary public sentiment and trends on key societal issues. Information. 2025;16(4):271.
8. Panja S. Information retrieval systems in healthcare: understanding medical data through text analysis. In: Transformative approaches to patient literacy and healthcare innovation. Hershey, PA, USA: IGI Global; 2024. p. 180–200.
9. Adithya K, Arulmoli R. Healthcare in your hands: sentiment analysis on Indian telemedicine in the era of global pandemic. In: Human machine interaction in the digital era. Boca Raton, FL, USA: CRC Press; 2024. p. 301–6.
10. Kayan F, Bilisli Y, Kayakus M, Açikgöz FY, Basdegirmen A, Güler M. Analysing sustainability and green energy with artificial intelligence: a Turkish English social media perspective. Sustainability. 2025;17(5):1882. doi:10.3390/su17051882.
11. Wang X, Sun F, Kim MG, Na HJ. Developing a novel audit risk metric through sentiment analysis. Sustainability. 2025;17(6):2460. doi:10.3390/su17062460.
12. Zhao H, Yang M, Bai X, Liu H. A survey on multimodal aspect-based sentiment analysis. IEEE Access. 2024;12:12039–52. doi:10.1109/access.2025.3624546.
13. Liu J, Li K, Zhu A, Hong B, Zhao P, Dai S, et al. Application of deep learning-based natural language processing in multilingual sentiment analysis. Mediterr J Basic Appl Sci. 2024;8(2):243–60. doi:10.46382/mjbas.2024.8219.
14. Kulebanova S, Prodanova J, Dedinec A, Sandev T, Wu D, Kocarev L. Media sentiment on air pollution: seasonal trends in relation to PM10 levels. Sustainability. 2024;16(15):6513.
15. Plutchik R. A general psychoevolutionary theory of emotion. In: Emotion: theory, research, and experience. New York, NY, USA: Academic Press; 1980. p. 3–33.
16. Russell JA. A circumplex model of affect. J Pers Soc Psychol. 1980;39(6):1161. doi:10.1037/h0077714.
17. Khan TA, Sadiq R, Shahid Z, Alam MM, Su'ud MBM. Sentiment analysis using support vector machine and random forest. J Inform Web Eng. 2024;3(1):67–75. doi:10.33093/jiwe.2024.3.1.5.
18. Naithani K, Raiwani YP. Realization of natural language processing and machine learning approaches for text-based sentiment analysis. Expert Syst. 2023;40(5):13114. doi:10.1111/exsy.13114.
19. Murthy G, Allu SR, Andhavarapu B, Bagadi M, Belusonti M. Text based sentiment analysis using LSTM. Int J Eng Res Tech Res. 2020;9(5):299–303. doi:10.17577/ijertv9is050290.
20. Trilla A, Alias F. Sentence-based sentiment analysis for expressive text-to-speech. IEEE Trans Audio Speech Lang Process. 2012;21(2):223–33. doi:10.1109/tasl.2012.2217129.
21. Hung LP, Alias S. Beyond sentiment analysis: a review of recent trends in text-based sentiment analysis and emotion detection. J Adv Comput Intell Intell Inform. 2023;27(1):84–95.
22. Alantari HJ, Currim IS, Deng Y, Singh S. An empirical comparison of machine learning methods for text-based sentiment analysis of online consumer reviews. Int J Res Mark. 2022;39(1):1–19. doi:10.1016/j.ijresmar.2021.10.011.
23. Acheampong FA, Wenyu C, Nunoo-Mensah H. Text-based emotion detection: advances, challenges, and opportunities. Eng Rep. 2020;2(7):12189.
24. Nandwani P, Verma R. A review on sentiment analysis and emotion detection from text. Soc Netw Anal Min. 2021;11(1):81. doi:10.1007/s13278-021-00776-6.
25. Bharti SK, Varadhaganapathy S, Gupta RK, Shukla PK, Bouye M, Hingaa SK, et al. Text-based emotion recognition using deep learning approach. Comput Intell Neurosci. 2022;2022(1):2645381. doi:10.1155/2022/2645381.
26. Yadollahi A, Shahraki AG, Zaiane OR. Current state of text sentiment analysis: from opinion to emotion mining. ACM Comput Surv. 2017;50(2):1–33.

27. Zucco C, Calabrese B, Agapito G, Guzzi PH, Cannataro M. Sentiment analysis for mining texts and social networks data: methods and tools. Wiley Interdiscip Rev Data Min Knowl Discov. 2020;10(1):1333. doi:10.1002/widm.1333.
28. Zhao J, Liu K, Xu L. Sentiment analysis: mining opinions, sentiments, and emotions. Cambridge, MA, USA: MIT Press; 2016.
29. Sánchez-Núñez P, Cobo MJ, De Las Heras-Pedrosa C, Pelaez JI, Herrera-Viedma E. Opinion mining, sentiment analysis and emotion understanding in advertising: a bibliometric analysis. IEEE Access. 2020;8:134563–76. doi:10.1109/access.2020.3009482.
30. Rout JK, Choo KKR, Dash AK, Bakshi S, Jena SK, Williams KL. A model for sentiment and emotion analysis of unstructured social media text. Electron Commer Res. 2018;18:181–99.
31. Kosti R, Alvarez JM, Recasens A, Lapedriza A. Context based emotion recognition using EMOTIC dataset. IEEE Trans Pattern Anal Mach Intell. 2019;42(11):2755–66. doi:10.1109/tpami.2019.2916866.
32. Cowen AS, Keltner D. Self-report captures 27 distinct categories of emotion bridged by continuous gradients. Proc Natl Acad Sci U S A. 2017;114(38):E7900–9. doi:10.1073/pnas.1702247114.
33. Saheb Jam G, Rhim J, Lim A. Developing a data-driven categorical taxonomy of emotional expressions in real-world human–robot interactions. In: Companion of the ACM/IEEE International Conference on Human–Robot Interaction. New York, NY, USA: ACM; 2021. p. 479–83.
34. Rudkowsky E, Haselmayer M, Wastian M, Jenny M, Emrich Š, Sedlmair M. More than bags of words: sentiment analysis with word embeddings. Commun Methods Meas. 2018;12(2–3):140–57. doi:10.1080/19312458.2018.1455817.
35. Wang Y. Research on the TF-IDF algorithm combined with semantics for keyword extraction in online news texts. J Intell Syst. 2023;33(2):245–60. doi:10.1515/jisys-2023-0300.
36. Dey RK, Das AK. Modified term frequency-inverse document frequency based deep hybrid framework for sentiment analysis. Multimed Tools Appl. 2023;82(21):32967–90. doi:10.1007/s11042-023-14653-1.
37. Du KL, Zhang R, Jiang B, Zeng J, Lu J. Foundations and innovations in data fusion and ensemble learning for effective consensus. Mathematics. 2025;13(4):587. doi:10.3390/math13040587.