International Journal of Computational and Experimental Science and ENgineering (IJCESEN)

Vol. 11-No.3 (2025) pp. 4669-4678
http://www.ijcesen.com

ISSN: 2149-9144

Research Article

Sentiment Analysis for Transliterated Hindi and Marathi Language using Machine Learning Approach

Rishikesh Janardan Sutar [1,2]*, Kamalakar Ravindra Desai[3]

[1]*Department of E&TC Engineering., Department of Technology, Shivaji University, Kolhapur, India*
[2]*Department of E&TC Engineering, SCTR's PICT, Pune, India*
*** Corresponding author's Email:** rishikeshjsutar@gmail.com **- ORCID:** 0009-0003-2822-2158

[3]*Department of E&TC Engineering, Bharati Vidyapeeth's College of Engineering, Kolhapur, India*
Email: krdesai2013@gmail.com**- ORCID:**0000-0002-0316-6722

Article Info:

DOI: 10.22399/ijcesen.3115
Received: 22 April 2025
Accepted: 25 June 2025

Keywords

Sentiment analysis
Hindi-Marathi Transliterated Text
Spelling Variations
Sentiment words dictionary
Lexical analysis

Abstract:

Sentiment analysis for local transliterated languages such as Hindi and Marathi has gained increasing research interest due to the linguistic diversity and informal nature of user-generated content. However, most existing approaches are limited by insufficient datasets that fail to capture the wide range of transliteration-based spelling variations inherent in such languages. To address this gap, the present study introduces a manually curated sentiment word dictionary for Hindi and Marathi, enriched with diverse transliterated spellings and associated sentiment weights. Using this resource, multiple sentence-level datasets were developed, including Hindi, Marathi, and real-world YouTube comment datasets, where each sentence is annotated with an average sentiment score derived from constituent sentiment words. A comprehensive sentiment classification framework was then designed using three feature extraction strategies: Count Vectorizer (CV), TF-IDF Vectorizer, and a Graph Embedding Technique (GET) combined with Rank-Based Selection (RBS). These features were used to train and evaluate three machine learning classifiers, Support Vector Machine (SVM), Logistic Regression (LR), and Random Forest (RF), which relies mainly on manually engineered linguistic features and graph-based representations. Experimental results demonstrate that SVM consistently outperforms LR and RF across all feature configurations. Among all combinations, SVM with TF-IDF achieved the highest accuracy, while SVM with GET+RBS demonstrated robust performance across datasets. Furthermore, the Hindi, Marathi, and mixed Hindi-Marathi datasets yielded comparable and higher accuracies than the YouTube comments dataset, confirming the advantage of structured transliterated corpora in sentiment analysis.

1. Introduction

In the era of digital communication, the proliferation of user-generated content on social media platforms has posed new challenges especially in countries like India, particularly for transliterated text. Transliterated text in Roman script is widely used on social media by bilingual Indian users, who speak but cannot write Hindi or Marathi, making sentiment analysis of such text an emerging research area [1]. Traditional NLP systems, which are optimized for monolingual and grammatically structured languages, often struggle to handle such transliterated content. Transliteration, the process of phonetically spelling native language words in a non-native script (often Roman), adds complexity to the sentiment analysis task.

The complexity of analysing such text especially arises from its inconsistent spelling. This challenge necessitates specialized approaches in the field of sentiment analysis technique used to automatically identify the emotional tone behind text.

While many researchers have explored sentiment analysis in native scripts or standardized Roman representations, there remains a gap in analysing transliterated Hindi and Marathi text using supervised machine learning on richly annotated, spell-varied datasets. Additionally, accuracy in a

sentence-based approach can be enhanced by expanding the sentiment dictionary with new words [2]. This research addresses the gap in sentiment analysis for Hindi and Marathi by first extracting sentiment-bearing words from standard bilingual dictionaries [3-4] to form a core vocabulary and then developing a novel transliteration-aware sentiment dictionary, evaluated using machine learning classifiers across various linguistic challenges.

The structure of this work includes related works (Section 2), the proposed methodology (Section 3), results (Section 4), discussion (Section 5), and conclusion with future scope (Section 6).

2. Related works

Sentiment analysis has gained prominence, especially for low-resource languages like Hindi and Marathi, due to the rise of social media and user-generated content. A major challenge is the use of Romanized (transliterated) scripts instead of native ones, which introduces variations in language and tone and a lack of labelled datasets. It reviews key studies, methods, and datasets addressing these challenges.

Ansari and Govilkar [5] created a classifier for transliterated Hindi-Marathi using models like Naïve Bayes, KNN, and SVM, showing ML's superiority over ontology-based methods. Srinivasan and Subalalitha [6] tackled class imbalance in code-mixed data using Levenshtein distance and classifiers such as Random Forest and XGBoost. Khare and Khan [7] analyzed SVM, Random Forest, and deep learning techniques for Hindi and Marathi, highlighting effective feature extraction. Pandey and Govilkar [8] found combining semantic features, WordNet, and SVM enhances accuracy in low-resource scenarios. Alam et al. [9] emphasized the need for domain adaptation and balancing noisy, code-mixed data. Sharma et al. [10] used lexicon-based and supervised learning methods for Hindi, stressing feature selection. Horvat et al. [11] proposed a hybrid rule-based and ML model for emotion detection in low-resource languages. Sidhu et al. [12] reviewed Hindi sentiment analysis, covering negation handling, lexicon approaches, and tools like stemmers. Chanda et al. [13] used language tagging and multilingual embeddings to improve sentiment classification in code-mixed Dravidian languages. Mulatkar and Bhojane [14] explored Hindi WordNet and SVM/Weka C4.5 models, addressing negation challenges. Shekhar et al. [15] introduced an artificial immune system with LSTM for ambiguous code-mixed data. Kumar et al. [16] explored hybrid and transformer-based models, focusing on contextual embedding and sarcasm detection. Rani and Kumar [17] used CNNs for Hindi movie reviews. Ahamad and Mishra [18] developed a unified ML model for both handwritten and digital text, including transliterated content. Sharma et al. [19] highlighted NLP's role in handling sarcasm and multilingual sentiment, especially in low-resource contexts. Sharma and Lakhwani [20] used the PRISMA method to review cross-domain sentiment analysis, addressing feature alignment and domain adaptation. Sazan et al. [21] evaluated TF-IDF vs. FastText for depressive text in Bangla, showing transferability to transliterated sentiment tasks. Yadav et al. [22] used sentiment lexicons and ML for Hindi news content. Shelke et al. [23] reviewed sentiment analysis for Indian languages, calling Marathi an under-resourced but promising language. Pawar and Mali [24] focused on Marathi sentiment analysis, stressing the need for custom lexicons and supervised learning. Gupta and Ansari [25] highlighted Hindi's online growth and the need for sentiment mining beyond English. Bhoir et al. [26] addressed preprocessing for transliterated text, including normalization and spelling correction. Lomte et al. [27] surveyed Marathi sentiment techniques, emphasizing linguistic resources like Marathi WordNet. Thorat et al. [28] noted Hindi's online growth and discussed dataset expansion and algorithm efficiency. Ranjan and Poddar [29] developed transliteration-aware spell-correction for abusive content detection, applicable to broader NLP tasks. Liu et al. [30] analyzed how transliteration enhances crosslingual alignment. Eusha et al. [31] used ML, deep learning, and transformers for Tamil and Tulu, showcasing the strength of transformer models in noisy, transliterated contexts.

In conclusion, despite notable progress, significant gaps remain in sentiment lexicons and datasets for transliterated text, especially for low-resource Indian languages like Hindi and Marathi. A recurring theme across existing literature is the urgent need for enriched, well-annotated datasets that address the unique challenges of transliteration, spelling variations, and the lack of standardized sentiment resources [1,2,12,23,25,28]. Many researchers emphasize the importance of developing language-specific resources to improve sentiment classification performance. In alignment with these insights, the major contribution of the present work lies in performing sentiment analysis using comprehensive, manually curated lexicons for Romanized Hindi and Marathi, enriched with extensive spelling variations and associated sentiment weights. These lexicons form the basis for generating sentence-level datasets

with computed sentiment scores, enabling robust training and evaluation of machine learning models. The study also underscores the importance of effective preprocessing and resource development in advancing sentiment analysis in low-resource, code-mixed contexts.

3. Proposed methodology

The proposed work was conducted using a machine equipped with Google Colab Pro. The implementation was carried out in Python, utilizing libraries such as Scikit-learn, Pandas, NumPy, and Matplotlib with Seaborn for machine learning and visualization.

Methodology consists of block schematic of proposed work, creation of Sentiment words Dataset with word sentiment score (WSS) and sentence database with average sentence sentiment score (AvgSSS), feature extraction techniques, classification techniques and performance metrics.

3.1 Block schematic of proposed work

The proposed framework for sentiment analysis in transliterated Hindi and Marathi languages begins with the extraction of sentiment-bearing words from standard Oxford Hindi-English dictionary and salaamchaus Marathi-English dictionary. These words form the core vocabulary for further processing.

Each identified sentiment word is manually annotated with a word sentiment score (WSS) on a scale (e.g., -3 to +3), based on its contextual sentiment strength. This human-curated annotation ensures high reliability and domain relevance.

Considering the lack of standard transliteration conventions, numerous spelling variations for each sentiment word are generated. This step significantly enhances the model's ability to handle real-world noisy and diverse transliterated inputs.

Using the enriched dictionary of transliterated spell variants, synthetic sentences are constructed. Special attention is given to ensure that each sentence includes at least two different sentiment words. Each sentence's average sentiment score (AvgSSS) is computed as the average of the word sentiment scores (WSS) of the sentiment words it contains. This results in a comprehensive and representative sentiment sentence dataset.

The sentence-level dataset is vectorized using traditional text feature extraction methods such as Count Vectorizer (CV) and TF-IDF Vectorizer. Additionally, a graph embedding technique (GET) is applied to capture contextual and structural relationships among words. Rank-based feature selection (RBS) is employed to retain the most informative features, improving classification efficiency and performance.

The refined features are used to train multiple classifiers including Logistic Regression (LR), Support Vector Classifier (SVC) and Random Forest (RF). Hyperparameter tuning is conducted using GridSearchCV to identify optimal model configurations that maximize predictive performance. The performance of each model is evaluated using Accuracy and confusion matrix.

This methodological pipeline is designed to address the unique challenges of transliterated sentiment analysis, particularly the variability in spelling and lack of standardized datasets, leading to more robust and accurate sentiment classification. The proposed framework for sentiment analysis in transliterated Hindi and Marathi languages is as illustrated in the block schematic shown in Fig. 1.

3.2 Sentiment words dataset creation

Transliterated word dataset is created by identifying carefully the sentiment words from the Oxford Hindi-English Dictionary (13,231 words) for Hindi

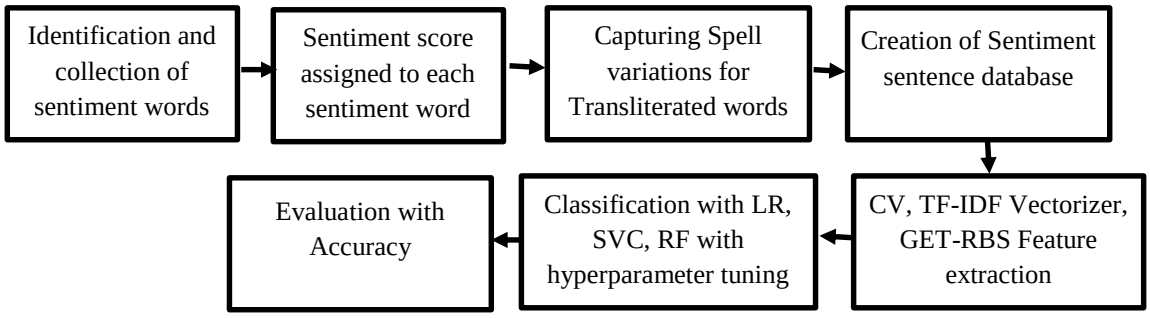

Figure 1. *Block schematic of proposed work*

language and from the salaamchaus's Chaus Marathi-English Dictionary (9,712 words) for Marathi language. Sentiment scores are assigned manually to each sentiment word, ranging from -3 to +3, with 0 as neutral, where, -3: most negative, -2: moderate negative, -1: less negative, 0: neutral, 1: less positive, 2: moderate positive, 3: most positive. Sentiment words with spelling variations are finally generated (176,755 spell varied words for Hindi and 159,804 spell varied words for Marathi).

From sentiment words dataset, one sample Marathi word with spell variations and assigned sentiment score is as below,

aanandee aanand: aanandee aanand2m+3, aannanddee aanand2m+3, anandi aanand2m+3, annanddi aanand2m+3, aannanddi aanand2m+3, aanandee aanand2m+3, aanandi aanand2m+3, annandi aanand2m+3, aannandee aanand2m+3, annanddee aanand2m+3, anandee aanand2m+3, ananddee aanand2m+3, annandee aanand2m+3, aananddee aanand2m+3, ananddi aanand2m+3, aannandi aanand2m+3, aananddi aanand2m+3

In "aanandee aanand2m+3", "aanandee aanand" is the sentiment word, "2" is for the number of sub-words from main sentiment word, "m" is for Marathi language, "+3" is the sentiment weight assigned for the sentiment word.

From sentiment words dataset, one sample Hindi word with spell variations and assigned sentiment score is as below.

anartha: anartha1h-2, annarrttha1h-2, anarta1h-2, annarrta1h-2, annarrtha1h-2, annarta1h-2, anarrttha1h-2, anarrta1h-2, annartta1h-2, anarttha1h-2, annartha1h-2, anartta1h-2, anarrtta1h-2, anarrtha1h-2, annarrtta1h-2, anartha1h-2, annarttha1h-2, anarth1h-2, annart1h-2, annartt1h-2, annarrt1h-2, anarth1h-2, anart1h-2, anartth1h-2, annarrth1h-2, annarrtth1h-2, anarrtt1h-2, anarrtth1h-2, annarth1h-2, annartth1h-2, anarrth1h-2, annarrtt1h-2, anarrt1h-2, anartt1h-2

In "anartha1h-2", "anartha" is the sentiment word, "1" is the number of sub-words from sentiment word, "h" is for Hindi language, "-2" is for sentiment weight assigned for sentiment word.

3.2 Sentence database creation

Sentence-level datasets (53,211 Hindi sentences and 30,659 Marathi sentences) are constructed from the sentiment words dataset. Special attention is given to ensure that each sentence includes at least two different sentiment words. Average sentence sentiment score i.e. AvgSSS for each sentence is calculated by the average word sentiment scores (WSS) of all the sentiment words found in that respective sentence as per Eq. (1).

$$AvgSSS = \frac{\sum_{i=1}^{n} WSS_i}{n} \quad (1)$$

where,

AvgSSS is average sentence sentiment score

WSS_i is the word sentiment score of ith word in that sentence

n is the total number of sentiment words in the sentence

The final sentiment score is ranging from -3 to +3, where,

Strongly positive, if $AvgSSS > 2$,

Moderately positive, if $1 < AvgSSS \leq 2$,

Less positive, if $0 < AvgSSS \leq 1$,

Neutral, if $AvgSSS = 0$,

Less negative, if $-1 \leq AvgSSS < 0$,

Moderately negative, if $-2 \leq AvgSSS < -1$,

Strongly negative, if $AvgSSS < -2$

Algorithm:

Algorithm for calculating sentiment score for each sentence in dataset is as shown below:

```
Load the sentence database
Calculate the number of sentences in database
Index the sentences as S1, S2, S3, S4, …. SN
For each sentence Si = 1 to N
    Read Si
    Calculate the length of Si
    Initialize Sentiment word count as SWC=0
    Initialize sentence sentiment score as SSS=0
    Split Si into words as w1, w2, w3, ….wn
    For each word wi = 1 to n
        If wi ϵ SentimentWordDictionary
            SWC = SWC +1
            Extract WSS
            SSS = SSS + WSS
        Else
            SSS = SSS
        If wi = wn
            AvgSSS = SSS / SWC
    End for
End for
```

In this way, it calculates the Average Sentiment score for each sentence in a sentence dataset and sentiment word count for each sentence. Few samples are shown in table 1.

3.3 Feature Engineering

To capture various facets of linguistic and contextual information, the following features are extracted:

3.2.1 Textual Features

Two classical vectorization methods are applied:

CountVectorizer (CV): Encodes text into sparse vectors based on raw word frequency.

Table 1. *Few samples of sentiment sentences with calculated sentence sentiment score (AvgSSS)*

Sentence preprocessed	Actual average sentence sentiment score	No. of Hindi words	No. of Marathi words	Rounded sentence sentiment score
wo sach mein khushnaseeb tha aur uski khushneeyat ne usse hamesha safalta dilayi	2.333333333	3	0	3
uska akrutadnya vyavhaar sabhi ko hairaan kar raha tha	-1.5	2	0	-2
akadoo vyakti ne apne akadhak vyavhaar se sabko hairaan kar diya	-1.333333333	3	0	-2
aamhaalaa paavan ani punyavaan lok bhetle	2	0	2	2
chhadmee lokanna chhal aawadte	-2	0	2	-2
premal svabhav premalapanaa dakhavto	3	0	2	3

TF-IDF Vectorizer: Encodes text by weighing term frequency with its inverse document frequency:

$$\text{tf-idf(t,d)} = \text{tf(t,d)} \cdot \log(\frac{N}{df(t)}) \quad (2)$$

where:
tf(t,d) is term frequency,
df(t) is document frequency,
N is the total number of documents.

3.2.2 Linguistic Features

The extracted numerical features are Average sentiment weight, Number of Hindi words, Number of Marathi words, Number of English words.
These are standardized using z-score normalization:

$$z = \frac{x-\mu}{\sigma} \quad (3)$$

3.2.3 raph-Based Embeddings (GET)

Sentences are treated as nodes in an undirected graph $G = (V, E)$, with edges linking sequential sentences. Node2Vec is applied to learn dense, low-dimensional node embeddings $e_i \in R^d$ through random walks and skip-gram optimization.

3.2.4 Rank-Based Feature Selection (RBS)

Redundant features are removed via VarianceThreshold.
Top k=50 features are selected using ANOVA F-statistic:

$$F = \frac{\sum_k n_k(\bar{x}_k-\bar{x})^2}{\sum_k \sum_i (x_{ki}-\bar{x}_k)^2} \quad (4)$$

3.3 Classification Models and Hyperparameter Tuning

The combined feature matrix (textual + numerical + graph) is split into training and test sets using an 80:20 ratio. Three classifiers are trained with hyperparameters tuned via GridSearchCV, and their best-performing settings are as follows:

3.3.1 Support Vector Machine (SVM)

Support Vector Machine (SVM) is employed in this study as one of the primary classification models for sentiment polarity prediction. The theoretical foundation of SVM lies in the maximization of the margin between two classes while minimizing classification errors. The optimization objective for a linear SVM is formally defined as:

$$\min_{w,b} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i \quad subject\ to$$
$$y_i(w^T x_i + b) \geq 1 - \xi_i, \xi_i \geq 0 \quad (5)$$

where 'w' is the weight vector, 'b' is the bias, 'ξ_i' are slack variables for misclassification, and 'C' is the regularization parameter that balances the trade-off between maximizing the margin and minimizing the classification error.
In practice, the above optimization is handled internally by the scikit-learn implementation of SVM. In our code, we utilized the 'SVC()' class from 'sklearn.svm', which abstracts this formulation and solves it using an appropriate quadratic programming solver.
To improve performance, hyperparameter tuning was carried out using GridSearchCV with 5-fold cross-validation. The following search space was defined:
param_grid= {'C': [0.1, 1, 10], 'kernel': ['linear', 'rbf']}
Best Parameters:
{'C'=10,'kernel'='linear'}
The input features to the SVM model were derived using:
Graph Embedding Techniques (GET): Using Node2Vec to embed sentence graphs into continuous vector space.
Numeric Linguistic Features: Including language-specific word counts and sentence weights.
Rank-Based Feature Selection (RBS): Using SelectKBest with ANOVA F-score to retain top 50 features.

3.3.2 Logistic Regression (LR)

Logistic Regression is employed as a linear classifier that predicts the probability of a sample belonging to a particular class using the sigmoid activation function. While the theoretical model is defined by the logistic (sigmoid) function:

$$P(y = 1|x) = \frac{1}{1+e^{-w^T x}} \quad (6)$$

In practice, this formulation was implemented using scikit-learn's 'LogisticRegression()' class, which internally optimizes the log-likelihood (cross-entropy loss) using numerical solvers such as L-BFGS and LibLinear.
In our experiments, hyperparameter tuning was performed using GridSearchCV over the

regularization parameter 'C' and solver type. The optimal configuration was selected based on 5-fold cross-validation accuracy:
param_grid = { 'C': [0.1, 1, 10, 100], 'solver': ['lbfgs', 'liblinear']}
Best Parameters:
{'C': 100, 'solver': 'lbfgs'}
The model was trained using the selected features from Graph Embedding Technique (GET) and Rank-Based Selection (RBS). The final classifier was evaluated on the test data using accuracy, confusion matrix, and classification metrics.
Probability model:
Trained using cross-entropy loss with L2 regularization.
Best Parameters:
{'C': 100, 'solver': 'lbfgs'}

3.3.3 Random Forest (RF)

Tree-based ensemble classifier. Splits based on Gini impurity:

$G=1-\sum_{i=1}^{C} p_i^2$ (7)

param_dist = {'n_estimators': [100, 200], 'max_depth': [None, 10], 'min_samples_split': [2, 5], 'min_samples_leaf': [1, 2]}
Best Parameters:
{'n_estimators': 200, 'max_depth': None, 'min_samples_split': 2, 'min_samples_leaf': 1}

3.6 Performance metrics

The models were evaluated using Accuracy as performance metrics:

Accuracy: Measures the proportion of correctly classified sentences out of the total sentences as per Eq. (8).

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (8)$$

where, TP is True Positive, FP is False Positive, TN is True Negative, and FN is False Negative.

4. Results:

The results are evaluated by model assessment with the help of accuracy.
Accuracy for all 54 combinations have been generated.

4.1 For Hindi Dataset: The following table 2 shows the results with various feature extraction methods and classification methods along with the hyperparameter tuning. Also, Fig. 2 shows comparative chart showing Accuracy with various Feature extraction techniques and classifiers for Hindi Dataset.

4.2 For Marathi Dataset: The following table 3 shows the results with various feature extraction methods and classification methods along with the hyperparameter tuning. Also, Fig. 3 shows comparative chart showing Accuracy with various Feature extraction techniques and classifiers for Marathi Dataset.

Table 2. *Accuracy and Confusion matrices for LR, SVC and RF with CV, TF-IDF and GET_RBS*

LR CV: 94.20 %	SVC CV: 96.92 %	RF CV: 89.01 %
LR TF-IDF: 85.15 %	SVC TF-IDF: 97.00 %	RF TF-IDF: 89.01 %
LR GET_RBS: 94.61 %	SVC GET_RBS: 96.99 %	RF GET_RBS: 86.95 %
LR CV GSCV: 94.60 % Best Parameters: {'C': 10, 'solver': 'lbfgs'}	SVC CV GSCV: 96.97 % Best Parameters: {'C': 10, 'gamma': 'scale', 'kernel': 'linear'}	RF CV GSCV: 90.38 % Best Parameters: {'n_estimators': 200, 'min_samples_split': 2, 'min_samples_leaf': 1, 'max_depth': None}
LR TF-IDF GSCV: 94.93 % Best Parameters: {'C': 100, 'solver': 'lbfgs'}	SVC TF-IDF GSCV: 97.00 % Best Parameters: {'C': 1, 'gamma': 'scale', 'kernel': 'linear'}	RF TF-IDF GSCV: 90.55 % Best Parameters: {'n_estimators': 200, 'min_samples_split': 2, 'min_samples_leaf': 1, 'max_depth': None}
LR GET_RBS GSCV: 94.78 % Best Parameters: {'C': 100, 'solver': 'lbfgs'}	SVC GET_RBS GSCV: 96.98% Best Parameters: {'C': 1, 'kernel': 'linear'}	RF GET_RBS GSCV: 94.60% Best Parameters: {'n_estimators': 200, 'min_samples_split': 2, 'min_samples_leaf': 1, 'max_depth': None}

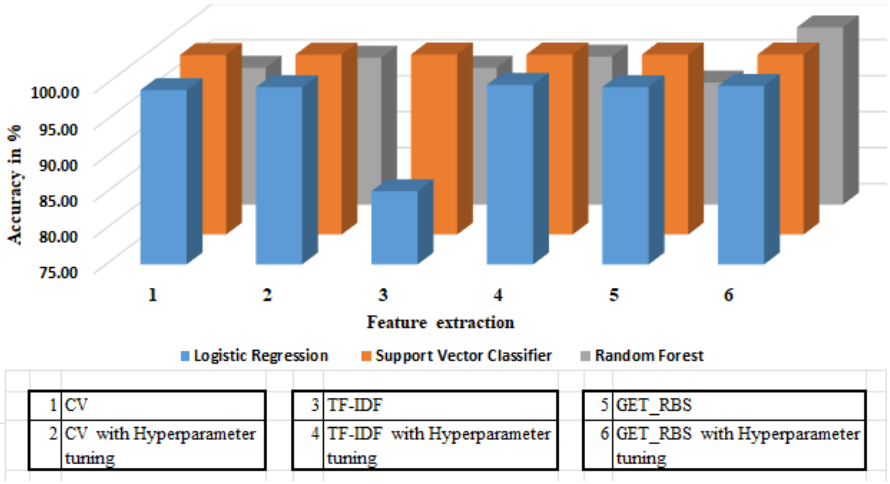

Figure 2. *Accuracy with various Feature extraction techniques and classifiers for Hindi Dataset*

Table 3. *Accuracy for LR, SVC and RF with CV, TF-IDF and GET_RBS*

LR CV: 94.61 %	SVC CV: 97.12 %	RF CV: 93.60 %
LR TF-IDF: 82.19 %	SVC TF-IDF: 96.00 %	RF TF-IDF: 93.21 %
LR GET_RBS: 94.67 %	SVC GET_RBS: 97.21 %	RF GET_RBS: 89.63 %
LR CV GSCV: 94.85 % Best Parameters: {'C': 100, 'solver': 'lbfgs'}	SVC CV GSCV: 96.24 % Best Parameters: {'C': 1, 'gamma': 'scale', 'kernel': 'linear'}	RF CV GSCV: 93.97 % Best Parameters: {'n_estimators': 200, 'min_samples_split': 2, 'min_samples_leaf': 1, 'max_depth': None}
LR TF-IDF GSCV: 94.93 % Best Parameters: {'C': 100, 'solver': 'lbfgs'}	SVC TF-IDF GSCV: 98.48 % Best Parameters: {'C': 1, 'gamma': 'scale', 'kernel': 'linear'}	RF TF-IDF GSCV: 93.60 % Best Parameters: {'n_estimators': 200, 'min_samples_split': 2, 'min_samples_leaf': 1, 'max_depth': None}
LR GET_RBS GSCV: 95.00 % Best Parameters: {'C': 100, 'solver': 'lbfgs'}	SVC GET_RBS GSCV: 98.72 % Best Parameters: {'C': 1, 'kernel': 'linear'}	RF GET_RBS GSCV: 94.41 % Best Parameters: {'n_estimators': 200, 'min_samples_split': 2, 'min_samples_leaf': 1, 'max_depth': None}

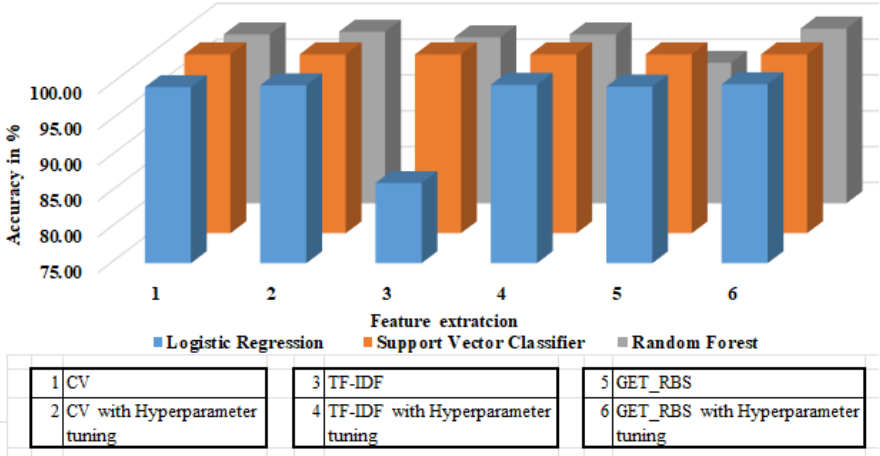

Figure 3. *Accuracy with various Feature extraction techniques and classifiers for Marathi Dataset*

4.3 For YouTube comments Dataset: The following table 4 shows the results with various feature extraction methods and classification methods along with the hyperparameter tuning. Also, Fig. 4 shows comparative chart showing Accuracy with various Feature extraction techniques and classifiers for YouTube comments Dataset.

Table 4. *Accuracy for LR, SVC and RF with CV, TF-IDF and GET_RBS*

LR CV: 89.96 %	SVC CV: 95.76 %	RF CV: 85.81 %
LR TF-IDF: 77.29 %	SVC TF-IDF: 96.53 %	RF TF-IDF: 86.58 %
LR GET_RBS: 93.26 %	SVC GET_RBS: 96.79 %	RF GET_RBS: 94.46 %
LR CV GSCV: 90.78 % Best Parameters: {'C': 10, 'solver': 'lbfgs'}	SVC CV GSCV: 95.76 % Best Parameters: {'C': 1, 'gamma': 'scale', 'kernel': 'linear'}	RF CV GSCV: 89.07 % Best Parameters: {'n_estimators': 100, 'min_samples_split': 2, 'max_depth': None}
LR TF-IDF GSCV: 94.33 % Best Parameters: {'C': 100, 'solver': 'lbfgs'}	SVC TF-IDF GSCV: 96.61 % Best Parameters: {'C': 10, 'kernel': 'linear'}	RF TF-IDF GSCV: 89.96 % Best Parameters: {'n_estimators': 100, 'min_samples_split': 2, 'max_depth': None}
LR GET_RBS GSCV: 95.40 % Best Parameters: {'C': 100, 'solver': 'lbfgs'}	SVC GET_RBS GSCV: 96.79 % Best Parameters: {'C': 1, 'kernel': 'linear'}	RF GET_RBS GSCV: 94.10 % Best Parameters: {'n_estimators': 200, 'min_samples_split': 10, 'min_samples_leaf': 1, 'max_depth': 20}

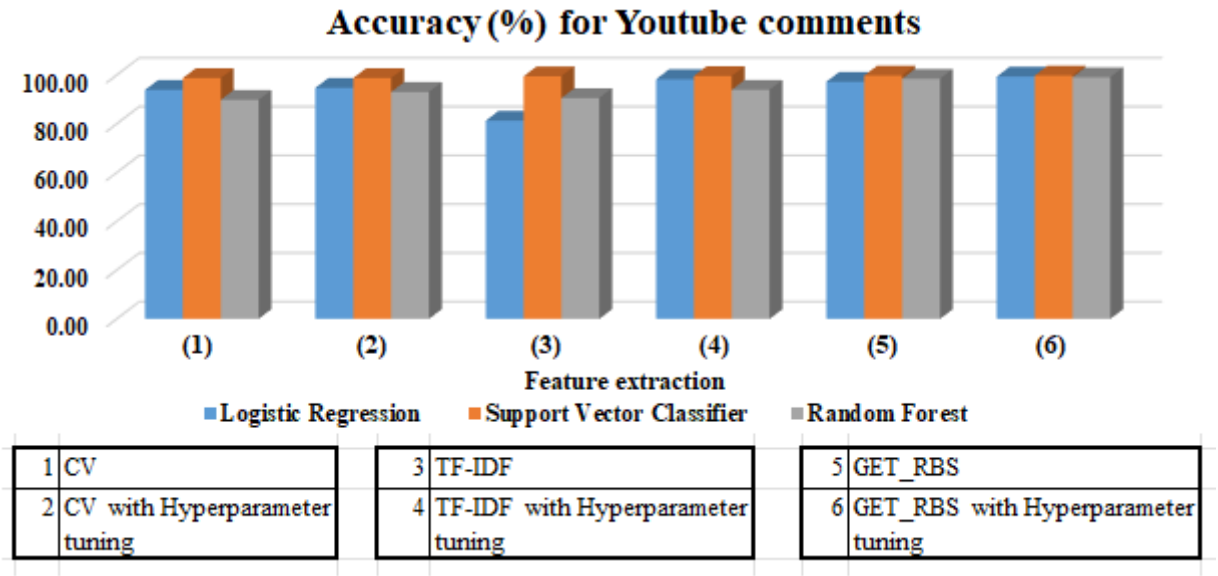

Figure 4. *Accuracy with various Feature extraction techniques and classifiers for Marathi Dataset*

5. Discussion:

The experimental analysis was conducted across multiple feature extraction strategies and classification models to evaluate sentiment polarity of multilingual sentence datasets. Notably, the combination of Support Vector Machine (SVM) with TF-IDF vectorization consistently achieved the highest classification accuracy, outperforming both Logistic Regression (LR) and Random Forest (RF) across datasets. This highlights the efficacy of SVM's ability to construct optimal separating hyperplanes in high-dimensional sparse feature spaces, which is particularly advantageous for text data represented through term frequency-based encodings.

In parallel, the Graph Embedding Technique (GET) using Node2Vec, when combined with Rank-Based Feature Selection (RBS), also yielded substantial performance improvements. This hybrid feature representation proved effective in capturing structural sentence relationships and language-specific attributes. While GET+RBS led to accuracy gains across all three classifiers (SVC, LR, RF), it was the SVC with GET+RBS that demonstrated better performance among these configurations. The robustness of SVM in handling high-dimensional and graph-augmented features likely contributed to this outcome.

Furthermore, Logistic Regression achieved moderate results, benefiting particularly from TF-IDF and GET-based features, but remained sensitive to the quality of embeddings and feature selection. On the other hand, Random Forest exhibited comparatively lower accuracy, likely due to its limited adaptability to sparse and dense mixed feature spaces such as TF-IDF and Node2Vec embeddings.

Across all dataset variations, it was observed that the Hindi and Marathi sentence datasets yielded comparable and higher accuracies relative to the YouTube comments dataset, which showed greater variability in structure, vocabulary, and informal expressions.

In conclusion, the findings confirm that SVC, particularly with TF-IDF and GET+RBS, offers a reliable and scalable solution for multilingual sentiment classification, outperforming conventional linear and ensemble classifiers across structured and graph-enhanced feature settings.

6. Conclusion with Future scope

The experimental results demonstrate that the combination of Support Vector Classification (SVC) with Graph Embedding Technique and Rank-Based Feature Selection (GET+RBS) yields the most effective performance for sentiment classification across all datasets used. This approach efficiently addresses the challenges posed by diverse non-standardized spellings through a manually curated sentiment lexicon enriched with transliteration variants. The use of graph-based embedding and statistical feature selection significantly contributes to enhanced model accuracy, establishing the robustness and reliability of the proposed framework in low-resource and code-mixed language contexts.

In addition to outperforming other models such as Logistic Regression and Random Forest, the SVC-based approach shows consistent accuracy across multiple dataset types, including Hindi and Marathi corpora, and showed comparable accuracy on noisy real-world YouTube comments. The findings affirm the suitability of the proposed methods for practical applications in multilingual, informal digital communication environments.

Future work can be directed towards extending the current framework in several directions. Firstly, the analysis may be expanded to include more complex linguistic phenomena such as sarcasm and free word order, which are prevalent in code-mixed and transliterated languages. Secondly, the sentiment lexicon development and spelling variation generation process demonstrated in this study can be generalized and applied to other low-resource Indian languages, thereby broadening the impact and applicability of this research. Also, comprehensive evaluation across various social media datasets and mixed-language corpora will further validate and strengthen the proposed methodology.

Author Statements:

- **Ethical approval:** The conducted research is not related to either human or animal use.
- **Conflict of interest:** The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper
- **Acknowledgement:** The authors declare that they have nobody or no-company to acknowledge.
- **Author contributions:** The authors declare that they have equal right on this paper.
- **Funding information:** The authors declare that there is no funding to be acknowledged.
- **Data availability statement:** The data that support the findings of this study are available on request from the corresponding author. The data are not publicly available due to privacy or ethical restrictions.

References

[1] M. Thomas, and C. Latha, (2020). "Sentimental analysis of transliterated text in Malayalam using recurrent neural networks", *Journal of Ambient Intelligence and Humanized Computing*, 2020, doi: 10.1007/s12652-020-02305-3.

[2] S. Deshmukh, N. Patil, S. Rotiwar, and J. Nunes, (2017). "Sentiment Analysis of Marathi Language", *International Journal of Research Publications in Engineering and Technology [IJRPET]*, Vol. 3, Issue 6, 2017, pp. 93-97.

[3] Oxford Hindi-English Dictionary

[4] Salaamchaus's Marathi-English Dictionary

[5] A. Ansari, and S. Govilkar, (2018). "Sentiment Analysis of Mixed code for the Transliterated Hindi and Marathi Texts", *International Journal on Natural Language Computing (IJNLC)*, Vol. 7, No.2, 2018, doi: 10.5121/ijnlc.2018.7202.

[6] R. Srinivasan, and C. Subalalitha, (2021). "Sentimental analysis from imbalanced code-mixed data using machine learning approaches", *Distributed and Parallel Databases*, doi: 10.1007/s10619-021-07331-4.

[7] B. Khare, and I. Khan, (2024). "Machine Learning Approaches for Sentiment Analysis in Hindi Text: A Comprehensive Survey", *International Journal of Innovative Research in Computer Science & Technology (IJIRCST)*, Vol. 12, Special Issue-1, 2024, doi: 10.55524/CSISTW.2024.12.1.62.

[8] P. Pandey, and S. Govilkar, (2015). "A survey of Sentiment Classification techniques used for Indian regional languages", *International Journal on Computational Science & Applications (IJCSA)*, Vol.5, No.2, 2015, pp. 13-26, doi:10.5121/ijcsa.2015.5202.

[9] [9] S. Alam, S. Mrida, and A. Rahman, (2025). "Sentiment Analysis in social media: How data science impacts public opinion knowledge integrates Natural Language Processing (NLP) with Artificial Intelligence (AI)", American *Journal of*

Scholarly Research and Innovation, *4*(01), 2025, pp. 63-100, doi: 10.63125/r3sq6p80.

[10] [10] S. Sharma, S. Bharti, and R. Goel, (2018). "A Frame Study on Sentiment Analysis of Hindi Language Using Machine Learning", *International Journal of Trend in Scientific Research and Development,* Vol. 2, 1603-1607, doi: 10.31142/ijtsrd14397.

[11] [11] M. Horvat, G. Gledec, and F. Leontić, (2024). "Hybrid Natural Language Processing Model for Sentiment Analysis during Natural Crisis", *Electronics 2024*, 13, 1991, doi: 10.3390/electronics13101991.

[12] [12] S. Sidhu, S. Khurana, M. Kumar, P. Singh, and S. Bamber, (2023). "Sentiment analysis of Hindi language text: a critical review", *Multimedia Tools and Applications*, 2023, doi: 10.1007/s11042-023-17537-6.

[13] S. Chanda, A. Mishra, and S. Pal, (2025). "Sentiment analysis of code-mixed Dravidian

[18] *and Engineering*, 2019, doi: 10.1007/s13369-018-3500-z.

[19] [18] R. Ahamad, and K. Mishra, (2025). "Exploring sentiment analysis in handwritten and E-text documents using advanced machine learning techniques: a novel approach", *J Big Data 12*, 2025, doi: 10.1186/s40537-025-01064-2.

[20] [19] N. Sharma, S. Ali, and A. Kabir, (2025). "A review of sentiment analysis: tasks, applications, and deep learning techniques", *Int J Data Sci Anal* 19, 351–388, 2025, doi: 10.1007/s41060-024-00594-x.

[21] [20] R. Sharma, and K. Lakhwani, (2024). "A Systematic Literature Review on Cross Domain Sentiment Analysis Techniques: PRISMA Approach", *Annals of Emerging Technologies in Computing (AETiC)*, Vol. 8, No. 4, 2024, doi: 10.33166/AETiC.2024.04.002.

[22] [21] S. Sazan, M. Miraz, and M. Rahman, (2024). "Enhancing Depressive Post Detection in Bangla: A Comparative Study of TF-IDF, BERT and FastText Embeddings", *Annals of Emerging Technologies in Computing (AETiC)*, Vol. 8, No. 3, 2024, doi: 10.33166/AETiC.2024.03.003.

[23] [22] O. Yadav, R. Patel, Y. Shah, and S. Talim, (2020). "Sentiment Analysis on Hindi News Articles", *International Research Journal of Engineering and Technology (IRJET),* Vol. 07 Issue: 05, 2020.

[24] [23] M. Shelke, and S. Deshmukh, (2020). "Recent Advances in Sentiment Analysis of Indian Languages", *International Journal of Future Generation Communication and Networking*, Vol. 13, No. 4, (2020), pp. 1656–1675.

[25] [24] S. Pawar, and S. Mali, (2017). "Sentiment Analysis in Marathi Language", *International Journal on Recent and Innovation Trends in Computing and Communication*, 2017, Vol. 5, Issue: 8, pp. 21-25.

[26] [25] S. Gupta, and G. Ansari, (2014). "Sentiment Analysis in Hindi Language: A Survey", *International Journal of Modern Trends in* languages leveraging pretrained model and word-level language tag," *Natural Language Processing*, Vol. 31, No. 2, pp. 477–499, 2025. doi:10.1017/nlp.2024.30.

[14] S. Mulatkar, and V. Bhojane, (2015). "Sentiment Classification in Hindi", *IOSR Journal of Computer Engineering (IOSR-JCE),* Vol. 17, Issue 4, 2015, PP 100-102, doi: 10.9790/0661-1741100102.

[15] S. Shekhar, D. Sharma, D. Agarwal, and Y. Pathak, (2020). "Artificial Immune Systems-Based Classification Model for Code-Mixed Social Media Data", *IRBM*, (2020), doi: 10.1016/j.irbm.2020.07.004.

[16] M. Kumar, L. Khan, and H-T Chang, (2025). "Evolving techniques in sentiment analysis: a comprehensive review", *PeerJ Comput. Sci. 11: e2592*, 2025, doi:10.7717/peerj-cs.2592

[17] S. Rani, and P. Kumar, (2019). "Deep Learning Based Sentiment Analysis Using Convolution Neural Network", *Arabian Journal for Science Engineering and Research (IJMTER),* Vol. 01, Issue 05, 2014, pp. 82-88.

[27] N. Bhoir, A. Das, M. Jakate, S. Lavangare, and D. Kadam, (2021). "A Study on Sentiment Analysis of Twitter Data for Devnagari Languages", *International Research Journal of Engineering and Technology (IRJET),* Vol. 08 Issue: 10, 2021.

[28] V. Lomte, P. Jadhav, O. Kalshetti, S. Deshmukh, and A. Jadhav, (2021). "Survey on Sentiment Analysis of Marathi Speech and Script", *International Research Journal of Engineering and Technology (IRJET),* Vol. 08 Issue: 12, 2021, pp. 876-893.

[29] M. Thorat, and N. Guide, (2022). "Review Paper on Sentiment Analysis for Hindi Language", *Grenze International Journal of Engineering and Technology, Jan Issue, Grenze Scientific Society*, 2022, Grenze ID: 01. GIJET.8.1.74.

[30] E. Ranjan, and N. Poddar, (2022). "Multilingual Abusiveness Identification on Code-Mixed Social Media Text", *arXiv:2204.01848v1 [cs.CL]*, 2022.

[31] Y. Liu, M. Wang, A. Kargaran, A. Imani, O. Xhelili, H. Ye, C. Ma, F. Yvon, and H. Schütze, (2024). "How Transliterations Improve Crosslingual Alignment", *arXiv:2409.17326*, 2024.

[32] A. Eusha, S. Farsi, A. Hossain, S. Ahsan, and M. Hoque, (2024). "Sentiment Analysis using Transformer-Based Models in Code-Mixed and Transliterated Tamil and Tulu", *Proceedings of the Fourth Workshop on Speech, Vision, and Language Technologies for Dravidian Languages*,2024, pp. 205–211.