Mocktails of Translation, Ensemble Learning and Embeddings to tackle Hinglish NLP challenges

Lov Kumar, Vikram Singh, Proksh, Pratyush Mishra
National Institute of Technology, Kurukshetra

Abstract

Social media has become a global platform where users express opinions on diverse contemporary topics, often blending dominant languages with native tongues, leading to code-mixed, context-rich content. A typical example is Hinglish, where Hindi elements are embedded in English texts. This linguistic mixture challenges traditional NLP systems, which rely on monolingual resources and need help to process multilingual content. Sentiment analysis for code-mixed data, mainly involving Indian languages, remains largely unexplored. This paper introduces a novel approach for sentiment analysis of code-mixed Hinglish data, combining translation, different stacking classifier architectures, and embedding techniques. We utilize pre-trained LoRA weights of a fine-tuned Gemma-2B model to translate Hinglish into English, followed by the employment of four pre-trained meta embeddings: *GloVe-T, Word2Vec, TF-IDF,* and *fastText*. SMOTE is applied to balance skewed data, and dimensionality reduction is performed before implementing machine learning models and stacking classifier ensembles. Three ensemble architectures, combining 22 base classifiers with a Logistic Regression meta-classifier, test different meta-embedding combinations. Experimental results show that the TF-W2V-FST (TF-IDF, Word2Vec, fastText) combination performs best, with SVM radial bias achieving the highest accuracy (91.53%) and AUC (0.96). This research contributes a novel and effective technique to sentiment analysis for code-mixed data.

1 Introduction

In communities where multiple languages are spoken, individuals often blend their native language with other prevalent languages during conversations. In digital space, including social media and texting with friends and family (Thara and Poornachandran, 2018), mixing languages emerges as a key force to steer a paradigm shift. Using more than one language in a single text, whether through code-mixing or code-switching, is a distinctive feature of social media communication. Social media and social networks (SMSN) platforms emerge as the key systems to encourage this contextual data.

Today, social media platforms are used for various activities, including keeping up with news, political and social events, sports, business, entertainment, and socializing (Kamwangamalu, 1989). Users also share reviews and opinions about products and services, often using multiple languages (Kim, 2006). This code-mixed text, written in another language's phonetic script, conveys strong sentiment and emotion, adding depth and authenticity. A significant portion of this text is in languages like Spanish, Chinese, Arabic, Hindi, and Urdu. The shift in opinion sharing has fueled interest in sentiment analysis to understand user content better. Initially, traditional data analytics provided insights into product reviews, trending topics, and targeted advertising. Now, NLP researchers focus on the complex code-mixed text, which includes spelling errors, hashtags, creative spellings (e.g., "b4" for "before"), abbreviations (e.g., "BTW" for "by the way"), phonetic typing (e.g., "becoz" for "because"), and wordplays (e.g., "gooood" for "good").

In recent years, users in multilingual countries like India have increasingly used native and Roman scripts to express their feelings (Parshad et al., 2016). With the growth of the internet and the expansion of user-generated content online, this practice has become increasingly prevalent in written text. For instance, **Hinglish Tweet**: *"@narendramodi ji, 2024 ke chunav mein phir se PM bante dekh khushi hui. Aapke netritva mein desh ka vikas aur unnati hoga."* and **English Translation**: "@narendramodi ji, It was a joy to see you become PM again in the 2024 elections. Under your leadership, the country will progress and prosper."

Sentiment analysis of code-mixed text is challenging (Srivastava and Singh, 2021) due to unstructured sentences, phonetic typing, mixed languages, spelling variations, and grammatical errors. This study builds on existing work by proposing a novel approach for Code-Mixed sentiment analysis for Hinglish. We utilize LoRA weights of fine-tuned LLM, fine-tuned on a code mixed (Hinglish-to-English) machine translation task, with different NLP & meta-embedding techniques.

1.1 Motivation and Research Questions (RQs)

Code-mixed text, standard on social media, is essential for sentiment analysis and emotion detection tasks. However, existing NLP tools struggle with the unique challenges of mixed-language data. This work addresses the complexities of Hinglish, a blend of Hindi and English prevalent on platforms like Twitter, especially in regions like India. This paper proposes a novel sentiment analysis framework for Hinglish tweets on X platform. We utilize LoRA weights of a fine-tuned LLM on a Hinglish-to-English machine translation task, combined with various NLP and meta-embedding techniques. We aim to predict the sentiment (positive or negative) of code-mixed tweets. This research aims to enhance NLP tools for code-mixed social media content, improving the analysis of emotions and opinions on products, politics, and events from social media. The conducted work is guided by the following research questions (RQs):

- **RQ1**: *How do various text representation techniques (e.g., TF-IDF, meta-embeddings) compare in their ability to capture semantics relevant to code-mixed sentiment classification when configured in different ways?*
- **RQ2**: *How do evaluation metrics like Accuracy and AUC differ across data balancing techniques in category prediction for code-mixed Hinglish data?*
- **RQ3**: *Can ensemble methods improve classification models' reliability and generalization efficiency, and how do different models compare in their performance?*

1.2 Contributions and Outline

The following outlines the contributions of this work:

1. The proposition of a novel sentiment analysis framework for code-mixed language 'Hinglish' tweets (from X platform) utilizing a fine-tuned Large Language Model for Hinglish-to-English translation as introductory work.
2. The proposed framework explores the various pre-trained meta-embedding techniques and their combinations in conjunction with an advanced sentiment analysis to deliver a pipeline. It thus utilizes combined efforts for the intended purpose.

2 Related Work

In recent work, Jadon et al. (Jadon et al., 2024) explored a hybrid LSTM-GRU model for sentiment analysis on Hinglish data, combining Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) architectures. This approach handled Hinglish's linguistic complexities effectively, achieving 96.76% accuracy. Frias et al. (Frias et al., 2023) examined Cross-lingual Word Embedding (CLWE) for sentiment analysis on a code-mixed Filipino-English corpus, developing a large, manually annotated feedback dataset on Higher Education Institutions. Using pre-trained transformer-based CLWE methods like mBERT, XLM-R, and XLM-T, they fine-tuned an Attention-based BiLSTM-CNN neural architecture. XLM-T achieved the highest performance with 91.30% accuracy, 90.36% precision, 90.92% recall, and a 90.61% F1-score.

Sampath and Supriya (Sampath and Supriya, 2024) introduced a method for translating code-mixed Hinglish, Malayalam-English, Tamil-English, and Telugu-English text into monolingual English using IndicLID for language identification and IndicTrans for transliteration and translation. IndicLID identified Indic languages with over 99% accuracy and code-mixed variants with 95%. Additionally, various ML and DL models were assessed for sentiment analysis on code-mixed data, with the DistilBERT tokenizer and classifier proving the most accurate. Similarly, Ansari and Govilkar (Ansari and Govilkar, 2018) proposed language identification at the word level along with POS tagging. Identified words were transliterated into native Indian languages (Hindi and Marathi), and sentiment scores were derived from SentiWordNet. NB and SVM classifiers were used, with the F1 scores for NB and Linear SVM outperforming those of RBF-based SVM for Hinglish.

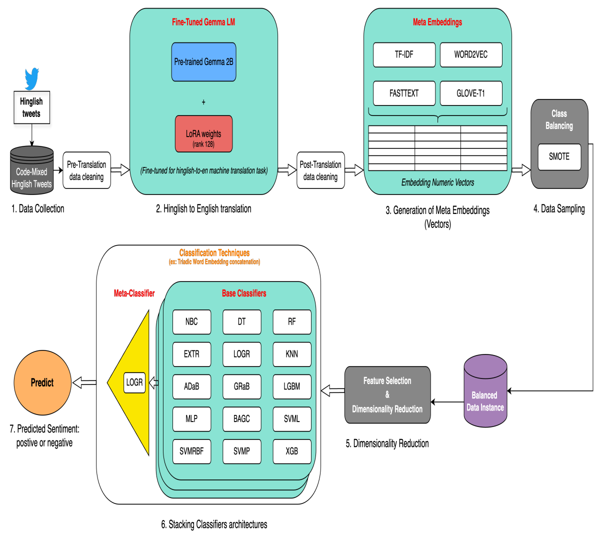
Singh and Lefever (Singh and Lefever, 2020) investigated sentiment analysis for Hinglish using unsupervised cross-lingual embeddings to capture word meanings across languages. This method, trained on SemEval 2020 data ((Patwa et al., 2020)), outperformed models trained on monolingual embeddings, achieving an F1-score of 0.635 compared to a baseline of 0.616. The cross-lingual embeddings facilitated transfer learning, allowing a sentiment model trained on English data to be applied to Hinglish data, resulting in an F1-score of 0.556. Singh (Singh, 2021) found that the highest F1-score of 0.6907 on the SemEval 2020 dataset was achieved using an Ensemble Voting (soft) ((Patwa et al., 2020)) classifier. This ensemble included SVM, Logistic Regression, and Random Forest, with the RF estimator parameter set to 750 and the SVM probability parameter set to true. The data was vectorized using a TF-IDF vectorizer with unigrams and a minimum occurrence frequency of 2.

To analyze Hinglish data, Sasidhar et al.(Sasidhar et al., 2020) created a dataset of 12000 Hindi-English code-mixed texts and annotated them with Happy, Sad, and Anger emotions. Then, a trained bilingual model was used to generate feature vectors, and deep neural models like 1D-CNN, LSTM, Bi-LSTM, CNN-LSTM, and CNN-BiLSTM were employed as classification models. They observed that the selected features, CNN-BiLSTM, performed best with 83.21% classification accuracy. Awatramani et al. (Awatramani et al., 2021) discussed the Lexicon-Based, Rule-Based, and ML approaches to study the effectiveness of classifying the text corpus with their appropriate sentiment labels. The Support Vector Machine (SVM) and Logistic Regression (LR) approaches gave the best results, with both algorithms giving an F1-score of 0.86 and an accuracy of 86%.

Gupta et al. (Gupta et al., 2021) introduced an unsupervised self-training framework for sentiment analysis of code-switched data using fine-tuned BERT models with pseudo labels from zero-shot transfer, achieving an F1-score of 0.32 and 36% accuracy for Hinglish. Mamta and Ekbal (Mamta and Ekbal, 2024) proposed a multilingual, multi-task model with a transformer-based pre-trained encoder for sentiment analysis of code-mixed and English texts. By incorporating English sentiment analysis as an auxiliary task, they fine-tuned the BERT encoder to capture shared and task-specific features, outperforming SOTA single-task systems on Hindi-English, Punjabi-English, and English datasets.

3 Study Design

Figure 1 illustrates the proposed pipelines of the work, with all the computational elements and interactions placed. The computational modules are designed using either of the fundamental techniques, e.g. *meta-embedding, data balancing, feature selection, feature scaling, or classification*. The proposed research study consists of two individual sets of modules: the Translation and Prediction modules. The translation module is a dedicated module for the '*Hinglish to English*' translation of code-mixed Tweets from Twitter, aka the 'X' platform. This module is prefixed and postfixed using data-cleaning techniques. Further, prediction modules take care of computations using different methods. The inherent details within each technique are provided subsequently.



Hinglish tweets
Code-Mixed Hinglish Tweets
Pre-Translation data cleaning
1. Data Collection
Fine-Tuned Gemma LM
Pre-trained Gemma 2B
+
LoRA weights (rank 128)
(Fine-tuned for hinglish-to-en machine translation task)
2. Hinglish to English translation
Post-Translation data cleaning
Meta Embeddings
TF-IDF
WORD2VEC
FASTTEXT
GLOVE-T1
Embedding Numeric Vectors
3. Generation of Meta Embeddings (Vectors)
Class Balancing
SMOTE
4. Data Sampling
Balanced Data Instance
Feature Selection & Dimensionality Reduction
5. Dimensionality Reduction
Classification Techniques
(ex: Triadic Word Embedding concatenation)
Meta-Classifier
LOGR
Base Classifiers
NBC
DT
RF
EXTR
LOGR
KNN
ADaB
GRaB
LGBM
MLP
BAGC
SVML
SVMRBF
SVMP
XGB
6. Stacking Classifiers architectures
Predict
7. Predicted Sentiment: postive or negative

Figure 1: Proposed Pipeline for Sentiment Prediction on Hinglish Code-Mixed Text.

3.1 Dataset Details

In this research, we utilized a dataset of tweets from the Mendeley Data repository (Pratibha et al., 2024) (dataset available at Mendeley Data). This data contains Hinglish tweets that vividly express raw emotions, sentiments, and textual gestures related to various conflict situations, with 410 and 984 for Positive and Negative sentiment labels, respectively. This includes, but is not limited to, *wars, crises, civil unrest, and world wars I, II,* etc.

1. **`tweets emotion 200`**: This file contains 195 rows of Hinglish tweets, each labelled with

one of 15 emotions, such as empathy, compassion, resilience, hope, gratitude, fear, anger, etc. The emoji field contains any emoji or emoticons if present in the tweet.

2. `emotions_tweets 1205`: This file contains 1,204 rows of Hinglish tweets, each labelled with one of 15 emotions: empathy, compassion, resilience, etc.

3.2 Data Preprocessing

The preprocessing steps are essential to ensuring the dataset's suitability and enhancing the overall performance of the sentiment analysis workflow. Three key processing steps are integrated: *Pre-translation data cleaning and Class grouping, translation of Hinglish text, and post-translation data cleaning and Normalization* elaborated below.

3.2.1 Sentiment Class Grouping and Pre-Translation Text Cleaning

Datasets contain 15 emotion classes in both files. These emotions are grouped into two major classes: positive and negative. `positive` class included classes such as 'empathy', 'compassion', 'resilience', 'hope', and 'gratitude', while `negative` class encompassed 'fear', 'anger', 'sadness', 'anxiety', 'shame', 'guilt', 'hopelessness', 'frustration', 'disgust', and 'grief'. A new column named `"Sentiment"` was created to store these respective binary labels, positive or negative. This binary classification facilitated an effective sentiment analysis. Further, the text cleaning within Hinglish tweets includes the removal of hashtags, Mentions, Special Characters and URLs, and Lowercasing. We have used regular expressions and pattern matching for this step. As a result of this thorough cleaning process for Hinglish tweets, the file `emotions_tweets 1205` lost four rows of data. Consequently, the total number of rows in the file decreased from 1204 to 1199. We then concatenated both files to make a single CSV file.

3.2.2 Translation of Hinglish Tweets and Post-Translation Cleaning

To handle code-mixed Hinglish tweets, we utilized pre-trained Low-Rank Adaptation (LoRA) weights of a fine-tuned high-performance LLM (available on the Hugging Face platform) to translate Hinglish to standard English. The model, fine-tuned on `gemma-2b` [1], employs PEFT (LoRA) with rank 128. This approach adapts the model's attention and feed-forward layers using low-rank matrices, allowing significant adaptation to Hinglish's unique syntactical and lexical characteristics with minimal parameter updates. Low-rank weight metrics occupy less storage space and provide similar performance.

We enabled LoRA on the GemmaCausalLM model with a rank of 128 and a sequence length of 256; loading pre-trained LoRA weights fine-tuned for Hinglish-to-English translation. A template with placeholders for Hinglish input and English output was used. To maintain grammatical integrity, we created a utility function to append periods to sentences lacking terminal punctuation. The translation process involved formatting the Hinglish sentence with the template, generating the English translation using the LoRA-enabled GemmaCausalLM model from the `keras_nlp` library, and extracting the translated text. We applied this translation function to each tweet in our dataset, creating a new column named 'Translated_Tweet' to store the English versions. This translation step was a critical component of our preprocessing pipeline, ensuring that our sentiment analysis approach could operate effectively on a uniform language basis, thus enhancing the accuracy and robustness.

The translated tweets underwent text cleaning and normalization using the Natural Language Toolkit (NLTK) for punctuation, special character removal, and stop-word removal. These preprocessing steps ensured the dataset was clean and consistent, enhancing the input data quality and significantly improving the machine learning model's performance in subsequent stages. Our choice of translating to English was influenced by the strengths of the employed LLM and embeddings in English, which allowed for higher accuracy and contextual understanding.

3.3 Mocktails of Word Embeddings Techniques

Word embeddings are critical in transforming textual data into a numerical format, enabling machine learning models to process and understand language effectively. We employed four pre-trained word embedding techniques, e.g. GloVe-T, Word2Vec (CBOW approach), TF-IDF, and fastText (character n-grams), in different combinations, aka Mocktail, e.g. TF-FST, TF-GL-FST, etc, to represent the textual data in a continuous vector space.

[1] `hf://google/gemma-2b-keras`

Our approach leverages meta-embedding learning, which involves generating a single (meta) word embedding from a set of pre-trained source word embeddings without pre-training the source embeddings or requiring access to the text corpora used to train them. Employed embeddings are computationally lightweight, require less computationally intensive resources.

3.4 Data Sampling and Feature Selection

To address the class imbalance, we used the Synthetic Minority Over-sampling Technique (SMOTE), which generates synthetic samples by interpolating between existing minority class samples. This method helps balance the class distribution and improves the model's learning from majority and minority classes.

For feature selection, we used a correlation-based dimensionality reduction approach. We removed highly correlated features by calculating correlation coefficients between feature pairs (absolute correlation coefficient ≥ 0.7) to reduce redundancy, eliminating multicollinearity and preserving the most informative features. We have also normalized these datasets using a Min-Max scaler, ensuring feature values ranged between 0 and 1, and removed columns with NaN values to maintain data integrity.

3.5 Classification Techniques and Validation

In the proposed pipeline, various stacking classifier configurations with different base learners were evaluated using cross-validation. We designed stacking classifiers with two to four base learners to enhance predictive performance, each operating on distinct meta-embedding feature spaces for diverse perspectives. The base learners included Naive Bayes (Gaussian, Bernoulli, Multinomial), Decision Trees, Logistic Regression, k-nearest Neighbors, and Support Vector Classifiers (linear, radial bias, polynomial kernel). Ensemble classifiers such as Bagging, Random Forest, Extra Trees, AdaBoost, Gradient Boosting, XGBoost, LightGBM, and Multi-Layer Perceptron (MLP) with different solvers (lbfgs, sgd, adam) were also employed. Logistic Regression served as the final estimator. Each stacking classifier incorporated two to four models from 22 machine-learning models trained on different meta-embedding features. For instance, Figure 2 shows a three-model stacking classifier using three distinct meta-embedding features for each classifier. Fifteen combinations of four embedding techniques were utilized, and the mentioned classifiers (excluding ensembles) were assessed for the single meta-embedding techniques. We used 3-fold cross-validation for robust performance evaluation, shuffling and splitting the data into training and testing sets.

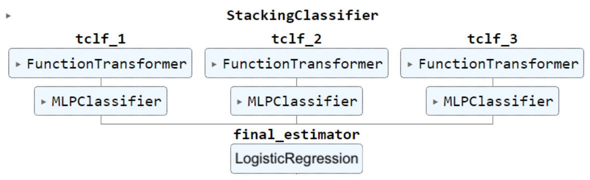


Figure 2: Instance of embedding Mocktails, as Triadic Word Embedding concatenation

4 Experimental Evaluation and Analysis

In the experimental analysis, the original dataset (ORGD) is transformed using 4 meta-embedding techniques, creating 4 new datasets. These datasets are balanced with SMOTE, resulting in 8 datasets. Dimensionality reduction is then applied, producing 16 datasets. We evaluate the predictive performance of various classifiers using our standalone meta-embedding techniques and three stacking architectures, combining various meta-embedding techniques with 2 to 4 classifiers from 22 ML models, using Logistic Regression as the meta-classifier. This process is applied to all 08 datasets, exploring 15 combinations of the 4 meta-embeddings for each of the 02 sets (SMOTE and ORGD). In total, 660 unique classification pipelines are created, representing different combinations of meta-embeddings, data balancing techniques, and classification algorithms. This comprehensive setup allows for a robust analysis of tweet categorization, enhancing the predictive modeling process.

The prediction pipelines are evaluated using Accuracy and AUC metrics, where accuracy indicates the proportion of correctly classified instances, and AUC measures the classifier's ability to distinguish between classes. Comparisons are made using box plots for AUC and accuracy, descriptive statistics (min, max, mean, median, Q1, Q3), and hypothesis testing using the Friedman and Wilcoxon Signed Rank Test.

4.1 Exploring the effect of Word-Embedding Techniques

The impact of embedding techniques on pipeline performance is evaluated using AUC and Accuracy, analyzed with box plots, descriptive statistics, and

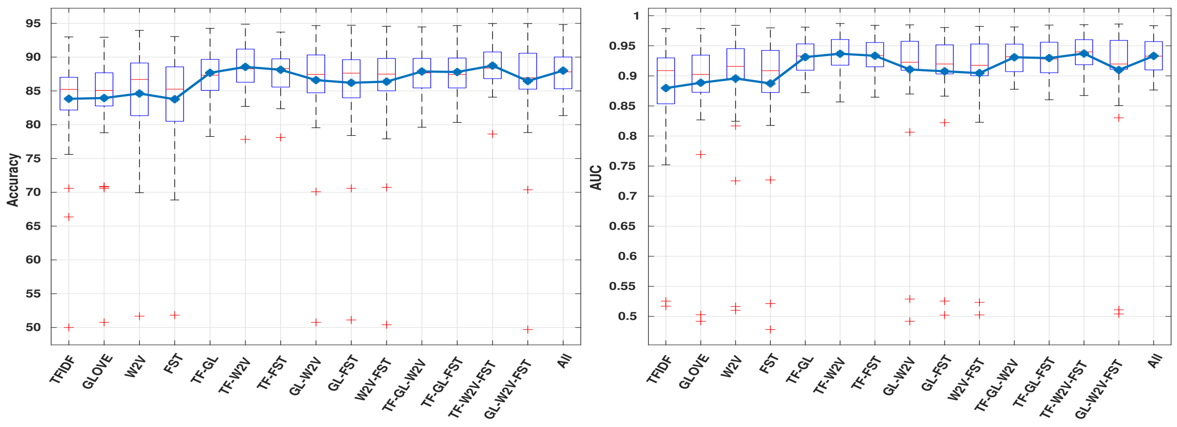


Figure 3: Performance box plots of employed Embedding techniques and Mocktails

hypothesis testing via the Friedman and Wilcoxon Signed Rank Test.

Descriptive statistic and Box-plots analysis: Figure 3 provides the box-and-whisker diagram of accuracy and AUC values. The stacking classifiers are trained on 15 combinations of our four original meta-embeddings. Figure 3 depicts that the models developed using the embedding technique TF-W2V-FST (TF-IDF, Word2Vec, and fastText) show the best performance with a mean AUC of 0.94, a median AUC of 0.94 and mean accuracy of 88.74%. TF-W2V technique shows similar performance.

Friedman test with Wilcoxon signed rank test: Two hypothesis tests have also been conducted on performance analysis: the Friedman and Wilcoxon Signed Rank Test with Bonferroni correction. First, the Friedman test has been used to examine the null hypothesis *"The performance values of stacking classifiers trained using twenty-two different classifiers show no significant improvement after applying different embedding techniques and their combinations"*. We rejected H_0 only if p ≤ 0.05.

Table 2 present the Friedman test results on Accuracy and AUC values, demonstrating that the models trained using different embedding techniques are significantly different. As indicated in Table 2, TF-W2V-FST secures the best mean rank on accuracy values of 3.18, and on AUC values, it is 3.30. This concludes that the technique TF-W2V-FST is best for developing the models for code-mixed sentiment analysis. Conversely, GLOVE has the highest mean rank, 13.02 on Accuracy and 12.80 on AUC, indicating that the model developed using standalone embeddings like GLOVE has the worst performance. After the Friedman test results, the study performed the Wilcoxon signed-rank test with Bonferroni correction. The considered hypothesis for the Wilcoxon test is *"The performance values of different embedding techniques when compared pairwise are significantly same"*. Table 2 shows the results of this test. This table shows that the performance of various word embedding techniques differs significantly for the task of code-mixed sentiment classification.

RQ1: How do various text representation techniques (e.g., TF-IDF, meta-embeddings) compare in their ability to capture semantics relevant to code-mixed sentiment classification when configured in different ways? ut of all the meta-embedding techniques and their combinations, TF-W2V-FST outperformed every other configuration with a mean AUC value of 0.94 and a mean AUC rank of 3.30.

4.2 Exploring the effect of Data Balancing Technique

The effectiveness of the SMOTE data balancing technique is evaluated by comparing accuracy and AUC, with results analyzed through box-and-whisker diagrams and hypothesis testing.

Descriptive statistic and Box-plots analysis: Figure 4 provides the box-and-whisker diagram of accuracy and AUC values. Figure 4 shows that the models trained on SMOTE-augmented data outperformed, with a mean AUC of 0.94, a median AUC of 0.95, and a mean accuracy of 87.96%. In comparison, models trained on original data achieved a mean AUC of 0.89 and a median AUC of 0.91. Thus, employing the SMOTE approach improved the AUC values of models from 0.89 to 0.94, representing a 5.62% enhancement in their predictive

Table 1: Hypothesis testing statistics of different Word Embedding and Mocktails.

	TFIDF	GLOVE	W2V	FST	TF-GL	TF-W2V	TF-FST	GL-W2V	GL-FST	W2V-FST	TF-GL-W2V	TF-GL-FST	TF-W2V-FST	GL-W2V-FST	All
TFIDF	1.00	0.63	0.23	0.64	0.00	0.00	0.00	0.01	0.03	0.08	0.00	0.00	0.00	0.02	0.00
GLOVE	0.63	1.00	0.28	0.98	0.00	0.00	0.00	0.02	0.05	0.11	0.00	0.01	0.00	0.03	0.00
W2V	0.23	0.28	1.00	0.31	0.05	0.01	0.03	0.14	0.37	0.57	0.08	0.10	0.01	0.19	0.06
FST	0.64	0.98	0.31	1.00	0.01	0.00	0.00	0.02	0.06	0.15	0.01	0.01	0.00	0.04	0.00
TF-GL	0.00	0.00	0.05	0.01	1.00	0.37	0.71	0.52	0.27	0.16	0.92	0.80	0.37	0.40	0.81
TF-W2V	0.00	0.00	0.01	0.00	0.37	1.00	0.58	0.19	0.06	0.03	0.29	0.25	0.99	0.15	0.45
TF-FST	0.00	0.00	0.03	0.00	0.71	0.58	1.00	0.41	0.17	0.09	0.63	0.52	0.52	0.34	0.82
GL-W2V	0.01	0.02	0.14	0.02	0.52	0.19	0.41	1.00	0.46	0.32	0.83	0.70	0.20	0.81	0.53
GL-FST	0.03	0.05	0.37	0.06	0.27	0.06	0.17	0.46	1.00	0.77	0.43	0.39	0.05	0.61	0.26
W2V-FST	0.08	0.11	0.57	0.15	0.16	0.03	0.09	0.32	0.77	1.00	0.24	0.23	0.03	0.39	0.13
TF-GL-W2V	0.00	0.00	0.08	0.01	0.92	0.29	0.63	0.83	0.43	0.24	1.00	0.89	0.25	0.71	0.72
TF-GL-FST	0.00	0.01	0.10	0.01	0.80	0.25	0.52	0.70	0.39	0.23	0.89	1.00	0.22	0.65	0.57
TF-W2V-FST	0.00	0.00	0.01	0.00	0.37	0.99	0.52	0.20	0.05	0.03	0.25	0.22	1.00	0.13	0.43
GL-W2V-FST	0.02	0.03	0.19	0.04	0.40	0.15	0.34	0.81	0.61	0.39	0.71	0.65	0.13	1.00	0.46
All	0.00	0.00	0.06	0.00	0.81	0.45	0.82	0.53	0.26	0.13	0.72	0.57	0.43	0.46	1.00
Accuracy	12.01	13.02	10.74	12.02	7.67	3.69	6.42	7.08	8.32	7.67	6.83	7.35	3.18	7.80	6.19
AUC	12.73	12.80	10.09	12.30	6.64	3.32	5.61	6.91	8.59	9.73	7.52	7.30	3.30	7.48	5.70

capability, with optimally balanced data to the prediction task.

Friedman test with Wilcoxon signed rank test: The same Friedman and Wilcoxon Signed Rank Test has been used here with Bonferroni correction to find the significant impact of using the sampling approach. Further, Table 4 lists the Friedman test values for SMOTE are 1.09 on accuracy and 1.01 on AUC, both lower than ORGD. Thus, models trained with SMOTE data are better for CMSA. A comparison of ORGD with SMOTE using the Wilcoxon signed rank test has been conducted following this test. Table 4 shows the result of this test, and we reject the hypothesis suggesting significantly improved performance on SMOTE-based prediction models than ORGD.

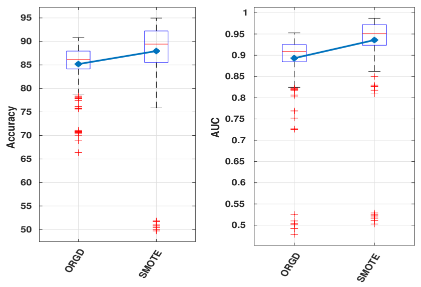


Figure 4: Performance box-plots of ORGD vs. SMOTE-based data.

Table 2: Hypothesis testing stats of Classes Imbalanced problem.

	ORGD	**SMOTE**
ORGD	1.00	0.63
SMOTE	0.63	1.00
Accuracy	1.91	1.09
AUC	1.99	1.01

***RQ2**: How do evaluation metrics like Accuracy and AUC differ across data balancing techniques in category prediction for code-mixed Hinglish data?* After applying SMOTE, the mean AUC value for ORGD, 0.89, improved to 0.94. Meanwhile, mean accuracy improved from 85.19% to 87.96%. Thus, balancing the data using SMOTE proved beneficial for CMSA task.

4.3 Exploring the performance of Classification Techniques

The stacking classifier architectures are validated with 3-fold cross-validation, and their effectiveness is assessed using Accuracy and AUC metrics, followed by hypothesis testing for further analysis.

Descriptive statistic and Box-plots analysis: Figure 5 shows the accuracy and AUC box plots for the different classifiers. The examination of Figure 5 concluded that among the general classifiers, the SVM with radial bias (SVCR) demonstrated superior performance, achieving a mean AUC of 0.96 and a mean accuracy of 91.53%. In contrast, the NB Bernoulli (BNB) classifier exhibited the lowest performance, with a mean AUC of 0.73 and a mean accuracy of 74%. Within the ensemble classifiers, LightGBM (LGBMC) performed the best with a mean AUC of 0.96 and a mean accuracy of 91.23%, closely followed by XGBoost (XGBC). However, AdaBoost (AdaB) and GradientBoosting (GRaB) displayed the lowest performance, with a mean AUC of 0.88. For MLP classifiers utilizing different solvers, the MLP with Adam solver (MLPA) outperformed, achieving a mean AUC of 0.94 and a mean accuracy of 89.56%. In contrast, the MLP with SGD solver (MLPS) had the lowest performance, with a mean AUC of 0.90 and a mean accuracy of 86.69%. Overall, the SVCR classifier emerged as the top performer, while LGBMC and

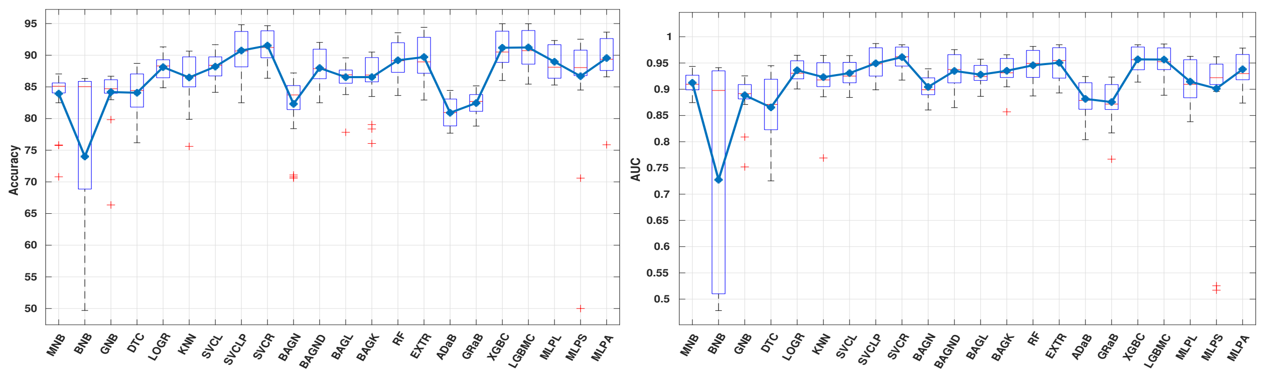


Figure 5: Performance Box-plots of employed Classifiers for Prediction.

XGBC performed similarly, securing second-best results. BNB classifier had the weakest performance.

Friedman test with Wilcoxon signed rank test: The Friedman test employs the Null hypothesis *'performance values of stacking classifiers show no significant improvement after changing training algorithms'*, and the summary listed in the Table 6 depicts that the stacking classifiers trained using different algorithms are not significantly similar. From Table 6, with a mean AUC rank of 2.00 and mean accuracy rank of 2.10, the analysis of the experiment concludes that SVCR performs best, while the GRaB classifier technique performs worst, with a mean AUC rank of 20.67 and mean accuracy rank of 19.78. AdaB offers a similar performance. After finding the best training algorithm, the investigation used the Wilcoxon signed rank test with Bonferroni correction to compare different training methods pairwise. The considered hypothesis for the Wilcoxon test is *"The performance values of stacking classifiers of different training algorithms, when compared pairwise, are significantly same"*. Table 6 shows the results of the Wilcoxon test with Bonferroni correction. Finally, it is observed that the SVCR outperforms significantly compared to other training methods. Therefore, the investigation recommends a classifier like SVCR to predict the sentiment of the code-mixed text.

RQ3**: **Can ensemble methods improve the reliability and generalization efficiency of classification models, and how do different models compare in their performance? As we saw, the TF-W2V-FST embedding approach outperformed others, demonstrating that a triad model ensemble is more effective than standalone classifiers. Among all the base classifiers, SVCR performed best with a mean AUC of 0.96 and an accuracy of 91.53%.

5 Conclusion

The paper exemplifies how LLMs, classifiers, embedding techniques, feature selection, and sampling can be combined effectively to enhance the performance and efficacy of predicting sentiment for code-mixed tweets. Accuracy and AUC are used to validate the effectiveness of each technique, while the Wilcoxon Sign Rank test and Friedman test statistically analyze the findings. The results conclude that for code-mixed Hinglish data, the TF-W2V-FST embedding approach achieved the best rank based on AUC and accuracy values, demonstrating that combining embedding techniques with ensembling outperforms standalone embeddings with general classifiers. The experimental findings also demonstrated improved performance after employing the SMOTE sampling technique. Additionally, the best mean rank for the CMSA task was obtained with the SVM radial bias (SVCR) classification algorithm, and LGBMC and XGBC both showed second-best results. The future directions may employ optimization methods to refine our research findings. Additionally, we aim to develop an LSTM-based approach for generating embeddings specifically tailored to Hinglish.

References

Mohammed Arshad Ansari and Sharvari Govilkar. 2018. Sentiment analysis of mixed code for the transliterated hindi and marathi texts. *Journal on Natural Language Computing (IJNLC)*, 7.

Priyanka Awatramani, Rucha Daware, Hrushabhsingh Chouhan, Anmol Vaswani, and Sujata Khedkar. 2021. Sentiment analysis of mixed-case language using natural language processing. In *2021 Third Interna-*

Table 3: Hypothesis testing of different ML classifiers.

	MNB	BNB	GNB	DTC	LOGR	KNN	SVCL	SVCLP	SVCR	BAGN	BAGND	BAGL	BAGK	RF	EXTR	ADaB	GRaB	XGBC	LGBMC	MLPL	MLPS	MLPA
MNB	1.00	0.01	0.00	0.00	0.00	0.03	0.00	0.00	0.00	0.11	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.82	0.04	0.00
BNB	0.01	1.00	0.50	0.21	0.00	0.00	0.00	0.00	0.00	0.15	0.00	0.00	0.00	0.00	0.00	0.42	0.47	0.00	0.00	0.00	0.00	0.00
GNB	0.00	0.50	1.00	0.17	0.00	0.00	0.00	0.00	0.00	0.03	0.00	0.00	0.00	0.00	0.00	0.09	0.04	0.00	0.00	0.03	0.00	0.00
DTC	0.00	0.21	0.17	1.00	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.52	0.83	0.00	0.00	0.00	0.00	0.00
LOGR	0.00	0.00	0.00	0.00	1.00	0.10	0.24	0.06	0.00	0.00	0.63	0.08	0.72	0.08	0.03	0.00	0.00	0.00	0.00	0.04	0.06	0.75
KNN	0.03	0.00	0.00	0.00	0.10	1.00	0.70	0.00	0.00	0.00	0.11	0.68	0.05	0.01	0.00	0.00	0.00	0.00	0.00	0.20	0.90	0.04
SVCL	0.00	0.00	0.00	0.00	0.24	0.70	1.00	0.01	0.00	0.00	0.28	0.51	0.21	0.03	0.01	0.00	0.00	0.00	0.00	0.14	0.52	0.17
SVCLP	0.00	0.00	0.00	0.00	0.06	0.00	0.01	1.00	0.12	0.00	0.07	0.01	0.06	0.46	0.89	0.00	0.00	0.33	0.27	0.00	0.00	0.08
SVCR	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.12	1.00	0.00	0.00	0.00	0.00	0.02	0.13	0.00	0.00	0.31	0.35	0.00	0.00	0.00
BAGN	0.11	0.15	0.03	0.01	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.35	0.00	0.00
BAGND	0.00	0.00	0.00	0.00	0.63	0.11	0.28	0.07	0.00	0.00	1.00	0.15	0.55	0.14	0.04	0.00	0.00	0.00	0.00	0.02	0.09	0.87
BAGL	0.00	0.00	0.00	0.00	0.08	0.68	0.51	0.01	0.00	0.00	0.15	1.00	0.07	0.01	0.01	0.00	0.00	0.00	0.00	0.31	0.75	0.13
BAGK	0.00	0.00	0.00	0.00	0.72	0.05	0.21	0.06	0.00	0.00	0.55	0.07	1.00	0.09	0.04	0.00	0.00	0.00	0.00	0.02	0.05	0.81
RF	0.00	0.00	0.00	0.00	0.08	0.01	0.03	0.46	0.02	0.00	0.14	0.01	0.09	1.00	0.36	0.00	0.00	0.09	0.09	0.00	0.01	0.21
EXTR	0.00	0.00	0.00	0.00	0.03	0.00	0.01	0.89	0.13	0.00	0.04	0.01	0.04	0.36	1.00	0.00	0.00	0.40	0.32	0.00	0.00	0.05
ADaB	0.00	0.42	0.09	0.52	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.51	0.00	0.00	0.00	0.00	0.00
GRaB	0.00	0.47	0.04	0.83	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.51	1.00	0.00	0.00	0.00	0.00	0.00
XGBC	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.33	0.31	0.00	0.00	0.00	0.00	0.09	0.40	0.00	0.00	1.00	0.81	0.00	0.00	0.00
LGBMC	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.27	0.35	0.00	0.00	0.00	0.00	0.09	0.32	0.00	0.00	0.81	1.00	0.00	0.00	0.00
MLPL	0.82	0.00	0.03	0.00	0.04	0.20	0.14	0.00	0.00	0.35	0.02	0.31	0.02	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.39	0.01
MLPS	0.04	0.00	0.00	0.00	0.06	0.90	0.52	0.00	0.00	0.00	0.09	0.75	0.05	0.01	0.00	0.00	0.00	0.00	0.00	0.39	1.00	0.07
MLPA	0.00	0.00	0.00	0.00	0.75	0.04	0.17	0.08	0.00	0.00	0.87	0.13	0.81	0.21	0.05	0.00	0.00	0.00	0.00	0.01	0.07	1.00
Accuracy	17.12	18.70	17.12	16.93	9.77	13.23	9.72	4.43	2.10	19.13	10.93	13.45	12.63	8.08	7.13	20.47	19.78	3.15	3.10	9.10	10.27	6.65
AUC	14.87	18.43	18.87	19.43	8.83	12.13	11.20	4.97	2.00	17.10	9.57	11.73	9.40	6.77	4.60	19.63	20.67	3.27	3.57	14.70	12.60	8.67

tional Conference on Inventive Research in Computing Applications (ICIRCA), pages 651–658. IEEE.

R. R. Frias, R. P. Medina, and A. M. Sison. 2023. Cross-lingual sentiment analysis of code-mixed corpus based on cross-lingual word embedding. In *2023 3rd International Conference on Intelligent Cybernetics Technology & Applications (ICICyTA)*, pages 101–106, Denpasar, Bali, Indonesia.

Akshat Gupta, Sargam Menghani, Sai Krishna Rallabandi, and Alan W. Black. 2021. Unsupervised self-training for sentiment analysis of code-switched data. *arXiv preprint arXiv:2103.14797*.

A. S. Jadon, M. Parmar, and R. Agrawal. 2024. Hinglish sentiment analysis: Deep learning models for nuanced sentiment classification in multilingual digital communication. In *2024 2nd International Conference on Device Intelligence, Computing and Communication Technologies (DICCT)*, pages 318–323, Dehradun, India.

Nkonko Mudipanu Kamwangamalu. 1989. Code-mixing and modernization. *World Englishes*, 8(3):321–332.

Eunhee Kim. 2006. Reasons and motivations for code-mixing and code-switching. *Issues in EFL*, 4(1):43–61.

Mamta and Asif Ekbal. 2024. Transformer based multilingual joint learning framework for code-mixed and english sentiment analysis. *Journal of Intelligent Information Systems*, 62(1):231–253.

Rana D. Parshad, Suman Bhowmick, Vineeta Chand, Nitu Kumari, and Neha Sinha. 2016. What is india speaking? exploring the "hinglish" invasion. *Physica A: Statistical Mechanics and its Applications*, 449:375–389.

P. Patwa, G. Aguilar, S. Kar, S. Pandey, S. Pykl, B. Gambäck, T. Chakraborty, T. Solorio, and A. Das. 2020. Semeval-2020 task 9: Overview of sentiment analysis of code-mixed tweets. *arXiv preprint arXiv:2008.04277*.

Pratibha, Amandeep Kaur, Meenu Khurana, and Robertas Damaševičius. 2024. Multimodal hinglish tweet dataset for deep pragmatic analysis. *Data*, 9(2):38.

Koyyalagunta Krishna Sampath and M. Supriya. 2024. Transformer based sentiment analysis on code mixed data. *Procedia Computer Science*, 233:682–691.

T. Tulasi Sasidhar, B. Premjith, and K. P. Soman. 2020. Emotion detection in hinglish (hindi+ english) code-mixed social media text. *Procedia Computer Science*, 171:1346–1352.

G. Singh. 2021. Sentiment analysis of code-mixed social media text (hinglish). *arXiv preprint arXiv:2102.12149*.

P. Singh and E. Lefever. 2020. Sentiment analysis for hinglish code-mixed tweets by means of cross-lingual word embeddings. In *Proceedings of the The 4th Workshop on Computational Approaches to Code Switching*, pages 45–51.

Vivek Srivastava and Mayank Singh. 2021. Challenges and limitations with the metrics measuring the complexity of code-mixed text. *arXiv preprint arXiv:2106.10123*.

S. Thara and Prabaharan Poornachandran. 2018. Code-mixing: A brief survey. In *2018 International conference on advances in computing, communications and informatics (ICACCI)*, pages 2382–2388. IEEE.