Poorvi@DravidianLangTech: Sentiment Analysis on Code-Mixed Tulu and Tamil Corpus

Poorvi Shetty
JSS Science and Technology University
Mysuru, India
poorvishetty1202@gmail.com

Abstract

Sentiment analysis in code-mixed languages poses significant challenges, particularly for highly under-resourced languages such as Tulu and Tamil. Existing corpora, primarily sourced from YouTube comments, suffer from class imbalance across sentiment categories. Moreover, the limited number of samples in these corpus hampers effective sentiment classification.

This study introduces a new corpus tailored for sentiment analysis in Tulu code-mixed texts. The research applies standard pre-processing techniques to ensure data quality and consistency and handle class imbalance. Subsequently, multiple classifiers are employed to analyze the sentiment of the code-mixed texts, yielding promising results. By leveraging the new corpus, the study contributes to advancing sentiment analysis techniques in under-resourced code-mixed languages. This work serves as a stepping stone towards better understanding and addressing the challenges posed by sentiment analysis in highly under-resourced languages.

Keywords sentiment analysis, code-mixed languages, Tulu, Tamil, under-resourced languages, corpus, class imbalance, classification.

1 Introduction

Online social media material is expanding at an exponential rate. Social media platforms allow users to freely express themselves in their native languages thanks to their multilingual user interface. As a result, a linguistic phenomenon known as code-mixing in social media data has become prevalent, attracting the interest of academics in disciplines like sociolinguistics and Natural Language Processing (NLP). The informality of code-mixed text, however, presents a number of difficulties, including those with data extraction and summarization. Sentiment analysis has been a significant research field in the field of code-mixed data analysis in recent years (Ahmad and Singla, 2021) (Patra et al., 2018) (Gambäck and Das, 2014) (Tarihoran and Sumirat, 2022).

India has the greatest population of speakers of English as a Second Language thanks to its rich linguistic past and close relationship with English. Native Indian language speakers don't use Unicode while exchanging information on social media platforms. They employ code-mixing to combine Latin script with English words or phrases in their original language to communicate themselves (Thara and Poornachandran, 2018). Additionally, there has been work done on sentiment analysis on YouTube comments (Alhujaili and Yafooz, 2021).

It is difficult to process these natural languages for diverse language-processing tasks (Srivastava and Singh, 2021). Compared to other languages, the regional languages of India are thought to have few resources (Harish and Rangan, 2020).

This paper introduces a novel methodology for sentiment analysis on code-mixed Tulu and Tamil corpora, considering the challenges associated with class imbalance. An additional corpus is curated by scraping YouTube comments on Tulu videos, enriching the code-mixed Tulu corpus and providing more comprehensive resources for analysis. This paper proposes a new stopwords list, tailored for both English and Tulu languages, and utilizes a synonyms list to address inconsistent spelling variations in the corpus. These contributions advance the field of sentiment analysis on code-mixed languages, offering insights and guidance for effective analysis and improving the overall performance of sentiment classification models.

2 Related Work

There is a lot of research being done right now on code mixing in natural language processing (NLP)

Third Workshop on Speech and Language Technologies for Dravidian Languages,
pages 124–132, Varna, Bulgaria, Sep 7, 2023.
https://doi.org/10.26615/978-954-452-085-4_016

jobs. In their thorough investigation of the difficulties code-mixed NLP faces in a multilingual society, Srivastava et al. (Srivastava and Singh, 2021) shed light on the current state of this field's NLP research.

The unique unified approach put out by Choudhary et al. (Choudhary et al., 2018) aims to overcome the drawbacks associated with using code-mixed text in NLP. Their method includes a preprocessing step that groups distinct word variations based on an empirical similarity measure, making analysis and processing more efficient.

Using datasets created expressly to show code-mixing between Bengali, English, and Hindi, Barman et al. (Barman et al., 2014) report an ongoing research project on automatic language recognition on social media platforms. According to their first results, a dictionary-based strategy outperforms supervised classification and sequence labelling techniques in solving this issue.

By creating a new code-switched dataset for Hindi-English language pairings and carrying out a comparative evaluation of conventional machine learning models for word-level language recognition, Mave et al. (Mave et al., 2018) make a contribution to the area. Their research offers insightful information about the performance of these models in contexts with code-mixed linguistics.

The following five language pairs underwent neural machine translation by Vyawahere et al (Vyawahare et al., 2022): Kannada to Tamil, Kannada to Telugu, Kannada to Malayalam, Kannada to Sanskrit, and Kannada to Tulu. The datasets for each of the five language pairings were used to train a variety of translation models, including Seq2Seq models like LSTM, bidirectional LSTM, Conv2Seq, and state-of-the-art transformers from scratch.

Although much research has been done on SA in the English language, data on the web also offers information in various other languages that should be examined. The goal of Shah et al. (Shah and Kaushik, 2019) is to analyse, assess, and debate the methodologies, algorithms, and difficulties encountered by the researchers when conducting the SA on Indigenous languages.

In today's age, Twitter contains a vast array of emotions and viewpoints. It offers a significant volume of sentiment-related information, but extracting data from Twitter necessitates appropriate techniques. The study conducted by Rakshitha et al. (Rakshitha et al., 2021) focuses on analyzing the sentiments expressed in regional languages on Twitter.

Labels	Train Set	Test Set
Positive	3487	344
Negative	736	60
Mixed Feelings	1094	107
Neutral	1921	197

Table 1: Tulu Corpus Details

Labels	Train Set	Test Set
Positive	22327	73
Negative	4751	338
Mixed Feelings	4458	101
Unknown State	6239	137

Table 2: Tamil Corpus Details

3 Corpus Details

In the case of the code-mixed Tulu corpus, a combination of the existing corpus (Hegde et al., 2022) and the newly created dataset proposed in this paper was utilized. For the code-mixed Tamil corpus, the data was obtained from the established dataset (Chakravarthi et al., 2020).

3.1 Existing Corpus

The code-mixed Tulu corpus used in this study was obtained from YouTube comments posted on videos in the Tulu language. These comments exhibited a mixture of languages including English, Kannada, Tulu, and combinations thereof, with varying scripts including Latin and Kannada. The comments in the corpus were manually labelled with sentiment categories, including positive, negative, neutral, and mixed feelings.

The code-mixed Tamil corpus utilized in this research was gathered from YouTube comments posted on videos in the Tamil language. The corpus predominantly consisted of "Tanglish" sentences, which are a combination of Tamil and English. It is noteworthy that the comments did not exclusively comprise fully Tamil or English sentences. To facilitate sentiment analysis, each comment in the corpus was annotated with sentiment labels, including positive, negative, unknown, and mixed feelings.

Labels	Number of Samples
Positive	323
Negative	65
Mixed Feelings	34
Neutral	195
Not Tulu	2932

Table 3: Newly-Created Tulu Corpus Details

Details	Number
Male, Post-graduate	1
Female, Graduate	1
Male, High-school graduate	1

Table 4: Details of Annotators for the Newly-Created Tulu Corpus

3.2 Creation of Additional Resources

3.2.1 New Corpus Creation for Tulu

To overcome the limited availability of samples in the original corpus, a supplementary dataset was curated by extracting comments from Tulu-language videos on YouTube. The YouTube Comment Scraper yielded a total of 3459 samples in this supplementary corpus. These comments were written using a combination of English, Kannada, and Tulu languages.

Notably, Tulu content is commonly articulated in either the Latin or Kannada scripts. I identified any comments not in the Tulu language and labelled them within the dataset manually.

Annotation Process: To ensure consistency, manual annotation was carried out on this new dataset, aligning with the sentiment categories in the corpus mentioned in the previous subsection. The sentiment classifications encompass Positive, Negative, Mixed Feelings, and Neutral tones.

Each sample is presented in a two-column format: 'Text,' which contains the full YouTube comment, and 'Annotations,' denoting the assigned sentiment category for the respective comment.

The annotation process engaged three native Tulu speakers, all proficient in English and Kannada also. The annotation guidelines closely followed the approach outlined by Hegde et al. After a preliminary demonstration featuring two comments from each sentiment class, the annotators independently assigned labels to all comments on their copy of the comment sheet.

For each individual sample, if the majority of annotators provided a consistent label, that label was selected. In instances of discordant labelling, the annotators collaborated in a discussion to reach a consensus.

This supplementary corpus plays a pivotal role in enhancing sentiment analysis tasks for the Tulu language. By augmenting the available dataset, it broadens the potential for more accurate sentiment classification. This corpus will be made available online.

Language	Number of Words
Tulu	193
English	127

Table 5: English-Tulu Stopwords List

3.2.2 Stopwords List Creation for Code-Mixed Tulu

To optimize the sentiment analysis process, a comprehensive list of English and Tulu stop words was meticulously compiled. The stop words in Tulu are presented in the Latin script to align with the character set commonly used in the comments.

The inclusion of stop words removal as a preprocessing step has demonstrated its efficacy in enhancing the performance of classification models in sentiment analysis tasks. (Sarica and Luo, 2021) By eliminating commonly occurring and less informative words, the focus is shifted towards more meaningful and sentiment-rich terms. To address the potential variations in spelling, special attention was given to accommodate different possible spellings of the same word. This consideration ensures a more robust and inclusive stop words list. The compiled list comprises a total of 320 words, encompassing both English and Tulu stop words. This resource facilitates the elimination of irrelevant and redundant terms.

3.2.3 'Synonyms' List Creation for Code-Mixed Tulu

Due to the absence of consistent spelling rules for Tulu in the Latin script, a challenge arises in dealing with the multiple spellings of the same word within the corpus. In response to this concern, with the aim of ensuring consistency in word usage, a meticulously crafted compendium of synonymous terms was developed for the textual corpus. This compilation was particularly attuned to words that have the highest frequencies in this corpus.

Each entry within the synonym list encompassed

Word	Synonyms/Variations in Spelling
super	superb, spr, supper, supr, sooper, sprb, superrrrr
malpule	manpule, malpi, malpere, malpuna, malpode, malpu, malpuni, maldar, malpad, malpun
panda	pand, pandh, pandat, panpar, pather, patherle, pande

Table 6: Sample Entries from 'Synonyms' List

terms sharing akin meanings, yet exhibiting variations in their orthographic representation. This compilation was methodically curated, and tailored exclusively to this specific dataset. It is not exhaustive and sought to collate words that convey identical meanings but had different spellings. Moreover, this compilation aimed to cluster words that share semantic equivalence, yet diverge in their linguistic structure and levels of respect. As an illustrative instance, the juxtaposition of "panper" and "panda" is encompassed within this list.

By incorporating these synonymous terms, the aim was to establish a standardised representation of commonly occurring words and minimize the impact of spelling inconsistencies on model performance.

The utilization of the synonym dictionary played a crucial role in enhancing the overall consistency of word spellings within the corpus. This, in turn, contributed to improved model performance in sentiment analysis tasks. By promoting uniformity in word representations, the dictionary of synonyms mitigated the challenges posed by varied spellings and facilitated more accurate sentiment classification.

4 Data Pre-Processing

The corpus underwent a series of pre-processing steps prior to the application of models. Initially, emojis were replaced with their corresponding names. Following this, a sequence of transformations was performed, which involved the removal of HTML tags, URLs, punctuation marks, special characters, numbers, and excessive whitespace. Additionally, the text was converted to lowercase to ensure uniformity.

Stop words, however, were not removed from the corpus, as their exclusion resulted in a decrease in performance.

To establish consistent spelling, a predefined list of synonyms that I created was utilized to replace words with their appropriate alternatives. Moreover, a label encoder was applied to the annotation labels, facilitating ease of use for the models during training (Shah et al., 2022). The TF-IDF Vectorizer was employed with specific parameters, including a maximum of 5000 features and an n-gram range of (1, 2) (Das and Chakraborty, 2018). In this context, the use of inverse document frequency (IDF) was disabled to optimize vectorisation.

To expand the corpus, the TextAttack EasyDataAugmenter technique was utilized, resulting in a quadrupling of the corpus size. This augmentation process helped to introduce additional variations in the data, thereby enhancing the overall model performance (Morris et al., 2020).

Considering the imbalanced distribution of positive comments within the corpus, the SMOTE (Synthetic Minority Over-sampling Technique) algorithm was employed. This technique ensured an equal representation of samples from each class, ultimately improving the performance of the models.

Class imbalance is a common challenge in many machine learning applications, particularly in sentiment analysis tasks. One effective approach to address this issue is Synthetic Minority Oversampling Technique (SMOTE). SMOTE is a data augmentation technique specifically designed to tackle class imbalance by generating synthetic samples for the minority class (Bowyer et al., 2011). The method works by identifying minority class instances and creating synthetic examples along the line segments connecting them. This process increases the diversity of the minority class and helps to balance the class distribution in the corpus. By introducing synthetic samples, SMOTE not only mitigates the impact of class imbalance but also improves the overall performance of classification models. It allows the classifier to learn from a more balanced representation of the data, leading to enhanced predictive capabilities. Moreover, SMOTE is widely applicable across various machine learning algorithms and has proven to be particularly effective in sentiment analysis tasks, where imbalanced sentiment classes are often encountered.

These pre-processing and data augmentation techniques collectively contributed to the refine-

ment and enrichment of the corpus, thereby facilitating more accurate sentiment analysis results.

5 Classification Models

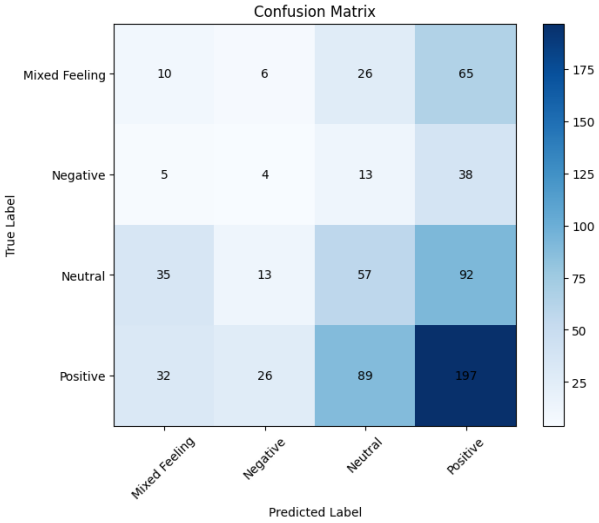

Figure 1: Confusion Matrix: Bagging Model on Tulu Dataset

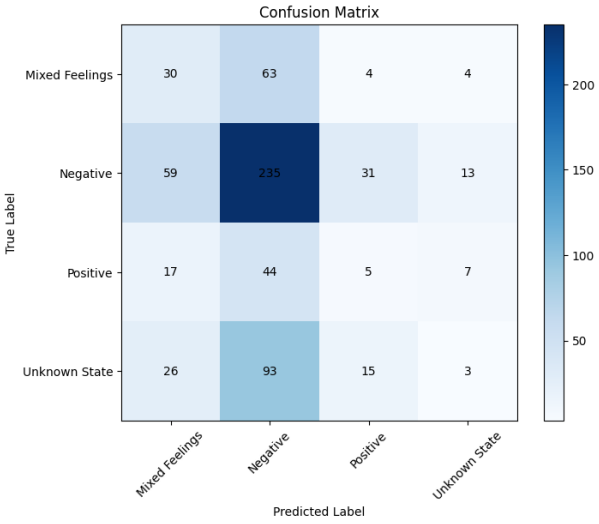

Figure 2: Confusion Matrix: AdaBoost Model on Tamil Dataset

In the study, a range of models (Pedregosa et al., 2018) were employed to analyze the corpus, and their parameters were fine-tuned using 5-fold cross-validation. The application of cross-validation ensures robust evaluation and provides insight into the models' performance. The chosen parameters aim to optimize the models' predictive capabilities while maintaining a balance between bias and variance. The details of the models considered in the analysis are given below.

5.1 Multinomial Naive Bayes

The Multinomial Naive Bayes classifier is a probabilistic model based on Bayes' theorem. It is specifically designed for classification tasks with discrete features, such as text classification. The alpha parameter, set to 0.1, represents the smoothing parameter that helps handle zero probabilities for unseen features.

5.2 Random Forest Classifier

Random Forest is an ensemble learning method that constructs multiple decision trees and combines their predictions to make final decisions. The criterion is set to 'entropy' to measure the quality of a split based on information gain. With a maximum depth of 8, the trees are limited in their growth to prevent overfitting. The max features parameter is set to 'log2' to control the number of features considered at each split. The n estimators parameter is set to 500, indicating the number of trees in the forest. The min samples split parameter is set to 7, determining the minimum number of samples required to split an internal node.

5.3 Logistic Regression

Logistic Regression is a linear classification model that estimates the probabilities of different classes. The C parameter is set to 10.0, controlling the inverse of the regularization strength. A higher C value indicates less regularization and a stronger emphasis on correctly classifying the training data. The max iter is set to 10000, defining the maximum number of iterations for the solver to converge. The penalty is set to 'l1', indicating L1 regularization that encourages sparse feature selection. The solver is set to 'liblinear', which handles L1 penalty efficiently.

5.4 Linear Support Vector Classifier (LinearSVC)

LinearSVC is a linear model for classification tasks based on Support Vector Machines (SVM). The max iter is set to 5000, defining the maximum number of iterations for convergence. The C parameter is set to 0.1, controlling the trade-off between margin maximization and misclassification. The penalty is set to 'l2', indicating L2 regularization that encourages small weights.

5.5 Decision Tree Classifier

The Decision Tree Classifier builds a tree model by recursively partitioning the data based on feature

Classifier	Mixed Feelings	Negative	Neutral	Positive	Macro Avg	Weighted Avg
Mutinomial NB	0.09	0.05	0.29	0.53	0.24	0.36
Random Forest	0.11	0.18	0.41	0.48	0.24	0.36
Logistic Regression	0.16	0.16	0.26	0.42	0.24	0.31
SVM	0.16	0.16	0.34	0.25	0.23	0.29
Decision Tree	0.06	0.03	0.42	0.06	0.14	0.16
KNN	0.25	0.10	0.29	0.02	0.17	0.14
AdaBoost	0.10	0.06	0.38	0.12	0.17	0.18
OneVsRest	0.16	0.07	0.30	0.39	0.23	0.30
XGBoost	0.16	0.10	0.36	0.29	0.23	0.27
GradientBoost	0.19	0.11	0.36	0.31	0.24	0.29
Voting	0.17	0.12	0.32	0.39	0.25	0.31
Stacking	0.06	0.04	0.28	0.49	0.22	0.33
Bagging	0.10	0.07	0.29	0.53	**0.25**	**0.36**

Table 7: F-Score for Tulu

Classifier	Mixed Feelings	Negative	Neutral	Positive	Macro Avg	Weighted Avg
Mutinomial NB	0.11	0.09	0.27	0.48	0.241	0.336
Random Forest	0.22	0.16	0.24	0.28	0.22	0.25
Logistic Regression	0.23	0.11	0.23	0.36	0.23	0.28
SVM	0.23	0.13	0.34	0.22	0.21	0.23
Decision Tree	0.04	0.10	0.88	0.09	0.16	0.18
KNN	0.25	0.13	0.26	0.01	0.16	0.13
AdaBoost	0.08	0.08	0.40	0.10	0.17	0.18
OneVsRest	0.22	0.07	0.26	0.33	0.22	0.27
XGBoost	0.25	0.11	0.37	0.08	0.20	0.19
GradientBoost	0.20	0.14	0.36	0.25	0.24	0.26
Voting	0.21	0.14	0.22	0.23	0.20	0.22
Stacking	0.14	0.11	0.29	0.42	0.24	0.31
Bagging	0.13	0.08	0.27	0.47	**0.243**	**0.336**

Table 8: F-Score for Tulu without 'Synonyms' List

Classifier	Mixed Feelings	Negative	Positive	Unknown	Macro Avg	Weighted Avg
Mutinomial NB	0.10	0.10	0.21	0.06	0.13	0.11
Random Forest	0.21	0.51	0.09	0.08	0.23	0.33
Logistic Regression	0.22	0.22	0.16	0.18	0.20	0.21
SVM	0.17	0.23	0.18	0.10	0.17	0.19
Decision Tree	0.07	0.66	0.02	0.02	0.19	0.36
KNN	0.01	0.02	0.20	0.02	0.06	0.04
AdaBoost	0.25	0.60	0.07	0.03	**0.24**	**0.37**
OneVsRest	0.16	0.22	0.18	0.15	0.17	0.19
XGBoost	0.26	0.46	0.12	0.10	0.23	0.32
GradientBoost	0.26	0.23	0.13	0.10	0.18	0.19
Voting	0.22	0.28	0.15	0.10	0.19	0.22
Stacking	0.13	0.12	0.20	0.04	0.12	0.11
Bagging	0.09	0.10	0.20	0.10	0.12	0.11

Table 9: F-Score for Tamil

values. The ccp alpha is set to 0.0001, representing the complexity parameter used for pruning the tree. The criterion is set to 'gini', which measures the impurity of a split. The max depth is set to 100, limiting the depth of the tree to avoid overfitting. The min samples split is set to 10, specifying the minimum number of samples required to split an internal node.

5.6 K-Nearest Neighbours Classifier (KNN)

K-Nearest Neighbors Classifier is a non-parametric algorithm that classifies samples based on their similarity to the k nearest neighbours. The n neighbours parameter is set to 1, indicating the closest neighbor is used for classification.

5.7 AdaBoost Classifier

AdaBoost is an ensemble method that combines weak classifiers into a strong classifier. The learning rate is set to 0.5, controlling the contribution of each weak classifier. The n estimators is set to 300, indicating the maximum number of estimators at which boosting is terminated.

5.8 One-vs-Rest Logistic Regression Classifier

The One-vs-Rest (OvR) strategy extends binary classifiers to multi-class classification. The logistic regression classifier (with C=10.0 and solver='liblinear') is used as the base classifier, and the OvR classifier combines multiple binary classifiers to handle each class.

5.9 Gradient Boosting Classifier

Gradient Boosting is an ensemble method that combines weak learners in a stage-wise manner, where each model tries to correct the errors made by the previous models. The n estimators is set to 50, indicating the number of boosting stages. The learning rate is set to 0.5, controlling the contribution of each weak learner. The max depth is set to 10, limiting the depth of each weak learner.

5.10 Voting Classifier

The Voting Classifier combines the predictions of multiple individual classifiers by a majority vote (hard voting). It includes three estimators: logistic regression, random forest, and linear support vector machine. The ensemble of these classifiers enables them to make joint decisions (Leon et al., 2017).

5.11 Stacking Classifier

The Stacking Classifier combines multiple classification models (k-nearest neighbours, random forest, and Multinomial Naive Bayes) by training a meta-classifier (Logistic Regression) on their predictions. This allows the meta-classifier to learn patterns from the outputs of the base classifiers and make the final prediction (Alexandropoulos et al., 2019).

5.12 Bagging Classifier

The Bagging Classifier applies the Bagging ensemble method to a base classifier (Multinomial Naive Bayes). It generates multiple subsets of the training data by bootstrapping and trains each subset on the base classifier. The final prediction is obtained through a majority vote of the base classifiers (Kotsiantis et al., 2005). These models and their respective parameters are applied to the corpus to explore their effectiveness in sentiment analysis tasks.

6 Experiments and Results

A series of experiments were conducted to identify the optimal configuration for sentiment analysis on the code-mixed corpus. The evaluation of the classification system's performance was based on the weighted averaged F-Score, which provides a comprehensive measure across all classes. To ensure reliable results, 5-fold cross-validation was employed to determine the best parameters for the models.

The corpus used for analysis exhibited an imbalance among the classes, necessitating the implementation of the Synthetic Minority Over-sampling Technique (SMOTE). This technique effectively addressed the class imbalance issue and led to significant improvements in the performance of most models.

Moreover, data augmentation techniques were employed using the TextAttack library. This approach further enhanced the corpus by generating additional samples, contributing to the overall performance improvement of the models.

After thorough experimentation and analysis, the stacking classifier, specifically the combination K-nearest neighbours, Random Forest, and Multinomial Naive Bayes with Logistic Regression as the meta-classifier, emerged as the best model for sentiment analysis on the code-mixed Tulu language. In contrast, logistic regression alone demonstrated

superior performance for sentiment analysis on the code-mixed Tamil language.

These findings highlight the effectiveness of the proposed models and the significance of addressing class imbalance and utilizing data augmentation techniques in code-mixed sentiment analysis tasks.

7 Conclusion

In conclusion, this paper presents a methodology for sentiment analysis on a code-mixed corpus consisting of Tulu and Tamil languages extracted from YouTube comments. The unique characteristics of code-mixed data, such as inconsistent spelling and the absence of stemming and lemmatisation libraries, pose challenges for traditional classifiers. This study looks at various classifiers and their performance of the code-mixed corpora. However, despite achieving notable performance, there remains ample room for further improvement in prediction accuracy. This study highlights the potential for future research endeavours to enhance sentiment analysis techniques specifically tailored for code-mixed languages.

8 Acknowledgements

I thank the organisers of Sentiment Analysis in Tamil and Tulu - DravidianLangTech@RANLP 2023 (Hegde et al., 2023) for giving me a platform to work on this topic.

References

Gazi Imtiyaz Ahmad and Jimmy Singla. 2021. Sentiment analysis of code-mixed social media text (sacmsmt) in indian-languages. In *2021 International Conference on Computing Sciences (ICCS)*, pages 25–33.

Stamatios-Aggelos Alexandropoulos, Christos Aridas, Sotiris Kotsiantis, and Michael Vrahatis. 2019. *Stacking Strong Ensembles of Classifiers*, pages 545–556.

Rawan Fahad Alhujaili and Wael M.S. Yafooz. 2021. Sentiment analysis for youtube videos with user comments: Review. In *2021 International Conference on Artificial Intelligence and Smart Systems (ICAIS)*, pages 814–820.

Utsab Barman, Amitava Das, Joachim Wagner, and Jennifer Foster. 2014. Code mixing: A challenge for language identification in the language of social media. In *Proceedings of the First Workshop on Computational Approaches to Code Switching*, pages 13–23, Doha, Qatar. Association for Computational Linguistics.

Kevin W. Bowyer, Nitesh V. Chawla, Lawrence O. Hall, and W. Philip Kegelmeyer. 2011. SMOTE: synthetic minority over-sampling technique. *CoRR*, abs/1106.1813.

Bharathi Raja Chakravarthi, Vigneshwaran Muralidaran, Ruba Priyadharshini, and John Philip McCrae. 2020. Corpus creation for sentiment analysis in code-mixed Tamil-English text. In *Proceedings of the 1st Joint Workshop on Spoken Language Technologies for Under-resourced languages (SLTU) and Collaboration and Computing for Under-Resourced Languages (CCURL)*, pages 202–210, Marseille, France. European Language Resources association.

Nurendra Choudhary, Rajat Singh, Ishita Bindlish, and Manish Shrivastava. 2018. Sentiment analysis of code-mixed languages leveraging resource rich languages.

Bijoyan Das and Sarit Chakraborty. 2018. An improved text sentiment classification model using TF-IDF and next word negation. *CoRR*, abs/1806.06407.

Björn Gambäck and Amitava Das. 2014. On measuring the complexity of code-mixing. In *Proceedings of the 11th international conference on natural language processing, Goa, India*, pages 1–7.

B. S. Harish and R. Kasturi Rangan. 2020. A comprehensive survey on indian regional language processing. *SN Applied Sciences*, 2(7).

Asha Hegde, Mudoor Devadas Anusha, Sharal Coelho, Hosahalli Lakshmaiah Shashirekha, and Bharathi Raja Chakravarthi. 2022. Corpus creation for sentiment analysis in code-mixed Tulu text. In *Proceedings of the 1st Annual Meeting of the ELRA/ISCA Special Interest Group on Under-Resourced Languages*, pages 33–40, Marseille, France. European Language Resources Association.

Asha Hegde, Bharathi Raja Chakravarthi, Hosahalli Lakshmaiah Shashirekha, Rahul Ponnusamy, SUBALALITHA CN, Lavanya S K, Thenmozhi D, Martha Karunakar, Shreya Shreeram, and Sarah Aymen. 2023. Findings of the shared task on sentiment analysis in tamil and tulu code-mixed text. In *Proceedings of the Third Workshop on Speech and Language Technologies for Dravidian Languages*, Varna, Bulgaria. Recent Advances in Natural Language Processing.

Sotiris Kotsiantis, George Tsekouras, and P. Pintelas. 2005. Bagging model trees for classification problems. pages 328–337.

Florin Leon, Sabina-Adriana Floria, and Costin Badica. 2017. Evaluating the effect of voting methods on ensemble-based classification. pages 1–6.

Deepthi Mave, Suraj Maharjan, and Thamar Solorio. 2018. Language identification and analysis of code-switched social media text. In *Proceedings of the Third Workshop on Computational Approaches to Linguistic Code-Switching*, pages 51–61, Melbourne, Australia. Association for Computational Linguistics.

John X. Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi. 2020. Textattack: A framework for adversarial attacks, data augmentation, and adversarial training in nlp.

Braja Gopal Patra, Dipankar Das, and Amitava Das. 2018. Sentiment analysis of code-mixed indian languages: An overview of sail$_{code}$ − $mixedsharedtask@icon$ − 2017.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Andreas Müller, Joel Nothman, Gilles Louppe, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2018. Scikit-learn: Machine learning in python.

Kakuthota Rakshitha, Ramalingam HM, M Pavithra, Advi HD, and Maithri Hegde. 2021. Sentimental analysis of indian regional languages on social media. *Global Transitions Proceedings*, 2(2):414–420. International Conference on Computing System and its Applications (ICCSA- 2021).

Serhad Sarica and Jianxi Luo. 2021. Stopwords in technical language processing. *PLOS ONE*, 16(8):1–13.

Deval Shah, Zi Yu Xue, and Tor M. Aamodt. 2022. Label encoding for regression networks.

Sonali Rajesh Shah and Abhishek Kaushik. 2019. Sentiment analysis on indian indigenous languages: A review on multilingual opinion mining.

Vivek Srivastava and Mayank Singh. 2021. Challenges and limitations with the metrics measuring the complexity of code-mixed text. In *Proceedings of the Fifth Workshop on Computational Approaches to Linguistic Code-Switching*, pages 6–14, Online. Association for Computational Linguistics.

Naf'an Tarihoran and Iin Ratna Sumirat. 2022. The impact of social media on the use of code mixing by generation z. *International Journal of Interactive Mobile Technologies (iJIM)*, 16(7):54–69.

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.

Aditya Vyawahare, Rahul Tangsali, Aditya Mandke, Onkar Litake, and Dipali Kadam. 2022. Pict@dravidianlangtech-acl2022: Neural machine translation on dravidian languages.