Cracking the Code: Machine Learning Approaches for Dravidian Code-Mixed Texts

Nikhil Narayan[1], Sachin Mohanty[1]

[1]Z-AGI Labs, India

Abstract

Word-level Language Identification (LI) is a crucial task in handling multilingual and code-mixed texts, where words from different languages appear in a single sentence, making it challenging to accurately identify the language of individual words. This task becomes even more complex when dealing with under-resourced languages such as Tulu, Kannada, Tamil, and Malayalam, where the availability of annotated datasets is limited. Recognizing this gap, the CoLI-Dravidian Shared Task@FIRE2024 was introduced by the organizers to address the need for comprehensive datasets and methods for word-level LI in code-mixed texts involving these under-resourced languages. To tackle this challenge, our team developed a robust methodology combining classical machine learning models—such as Logistic Regression, Support Vector Machines (SVM), Multinomial Naive Bayes, K-Nearest Neighbors (KNN), Decision Tree, and Random Forest—with advanced models like LightGBM (LGBM), CatBoost, and XGBoost. For each model, we performed extensive hyperparameter tuning to optimize performance and obtain the best possible scores. These models were trained using various embeddings, including Count Vectorizer and TF-IDF with n-gram ranges (1,3) and (1,4), as well as FastText embeddings, to effectively capture linguistic variations in the data. As a result, we achieved impressive rankings, securing Rank 4 in Kannada, Rank 3 in Malayalam, Rank 3 in Tulu, and Rank 2 in Tamil, demonstrating the effectiveness of our approach in this shared task.

Keywords

Low-Resource Languages, Code-Mixed Text, Dravidian Languages, Hyperparameter Optimization, Text Embeddings

1. Introduction

India's linguistic diversity is vast, with over 650 languages spoken across the country. The Dravidian language family, predominant in southern India, includes languages such as Tulu, Kannada, Tamil, and Malayalam. These languages are not only primary modes of communication for millions but also repositories of rich cultural and literary traditions. For example, Tamil boasts a literary history spanning over two millennia, while Malayalam has made significant contributions to poetry, drama, and philosophical discourse. Despite their cultural significance, Dravidian languages are underrepresented in digital technologies, particularly in Natural Language Processing (NLP) applications[1]. This underrepresentation poses challenges for integrating these languages into modern computational frameworks, limiting their availability in applications such as machine translation, sentiment analysis, and other AI-driven tools.

A significant challenge in processing these languages is word-level language identification (LI), especially in code-mixed texts where multiple languages appear within a single sentence. Code-mixing is prevalent on social media platforms, where speakers blend their native languages with English, creating a complex linguistic landscape that complicates text analysis[2]. This complexity is further exacerbated by the use of Roman script to phonetically transcribe these languages, resulting in mixed-script text that is difficult to process using conventional NLP tools[3]. Consequently, accurately identifying the language of individual words in such contexts becomes crucial for effective downstream NLP tasks.

The necessity of word-level LI arises from the growing need for accurate linguistic processing in multilingual environments. Without effective word-level LI, code-mixed texts become difficult to interpret, leading to reduced accuracy in applications like machine translation and sentiment analysis[4].

Forum for Information Retrieval Evaluation, December 12-15, 2024, India

nikhilnarayan73@gmail.com (N. Narayan); sachinmohanty269@gmail.com (S. Mohanty)

Moreover, the scarcity of annotated datasets for under-resourced languages further exacerbates the difficulty in developing robust models capable of handling the nuances of these linguistic combinations.

Given these challenges, the CoLI-Dravidian Shared Task@FIRE2024[5, 6] was established to advance research in word-level LI for code-mixed texts involving Dravidian languages. This shared task introduces annotated datasets comprising Tulu, Kannada, Tamil, and Malayalam texts, which are intricately blended with English and other local languages. These datasets, collected from user-generated content on social media, include categories such as 'Mixed-language,' 'Name,' 'Location,' and other specific linguistic markers, reflecting the nuanced and dynamic nature of language use in digital spaces. By providing a robust corpus for word-level LI, the task aims to enhance the development of models that can accurately identify and classify words in these diverse linguistic environments.

The development of such models is crucial for the broader field of NLP, as they offer the potential to improve various applications, including machine translation, sentiment analysis, and speech recognition, particularly for under-resourced languages. Addressing the complexities of code-mixed text will enable more accurate and inclusive NLP tools, fostering better understanding and processing of multilingual content in digital media.

From here, the report continues in the following manner: In section 2, we give an overview of the dataset for each language and describe the challenge at hand. In section 3, we present our approach in detail, covering the nitty gritty of our experimental set-up, cross-validation strategy, models used, and intuition behind them. In section 4, we brief the results from the experiments section. Then, we conclude in section 5 with the final takeaways, our standings, and the scope of future work.

2. Dataset Description

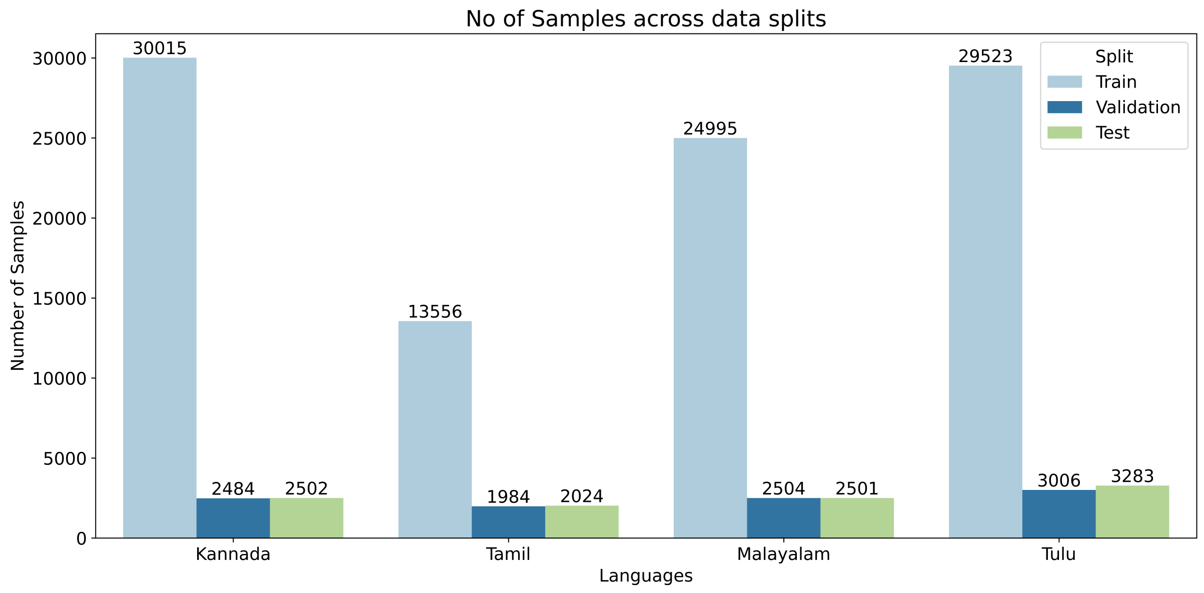

Figure 1: Train-Val-Test Sample Distrbution for different Languages

2.1. Dataset Overview

The dataset used for word-level Language Identification (LI) in code-mixed text spans four distinct subtasks based on different Indian languages: Tulu, Kannada, Tamil, and Malayalam. Each subtask focuses on LI in sentences containing a mix of English and a regional Indian language, presenting unique challenges and insights into the linguistic diversity of code-mixed text. Below is a detailed description of each subtask dataset and its composition, drawn from the competition website.

2.2. Tulu

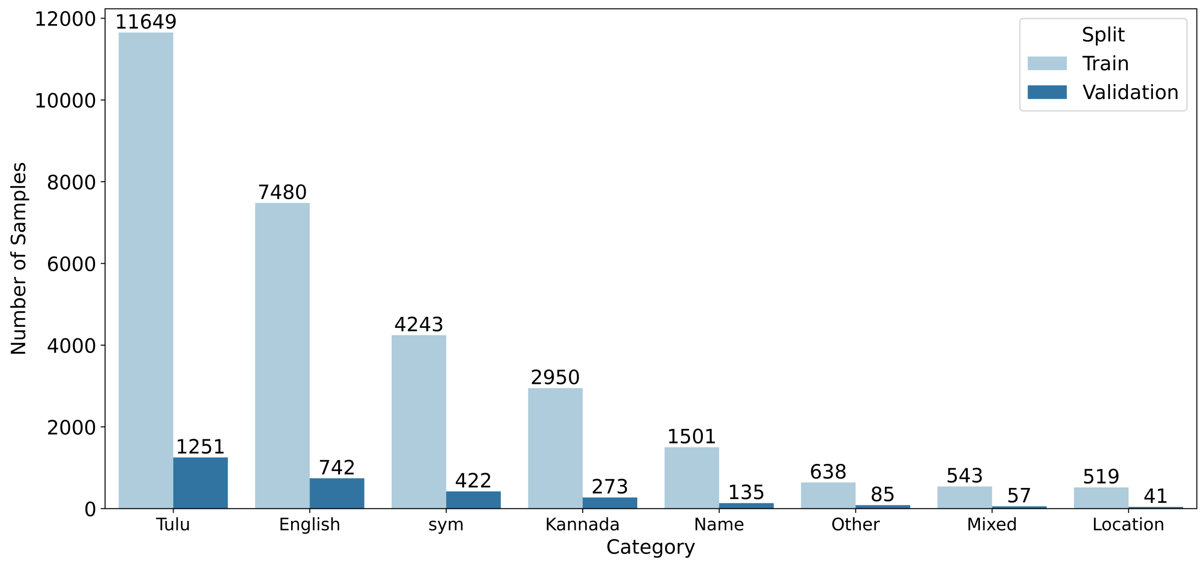

Figure 2: Train-val Target Distribution for Tulu language.

The Tulu dataset[7] comprises code-mixed sentences sourced from YouTube videos, which have been preprocessed to remove non-textual characters and standardized by transliteration into Roman script. The dataset categorizes the extracted words into classes including 'Tulu,' 'Kannada,' 'English,' 'Mixed-language,' 'Name,' and 'Location.' The complexity of the 'Mixed-language' class, which blends Tulu with Kannada and/or English, highlights the intricate linguistic patterns present in digital communication. This dataset is essential for developing robust models for language identification (LI) in Tulu-English-Kannada code-mixed text.

2.3. Kannada

The Kannada dataset[8] contains tokens that have been transliterated into Roman script to ensure uniform processing. These tokens are classified into 'Kannada,' 'English,' 'Mixed-language,' 'Name,' 'Location,' and 'Other.' This dataset is designed to enhance language identification in Kannada-English code-mixed texts, commonly found in informal digital communication. The 'Other' category provides additional flexibility by including tokens that do not align with the main language categories.

2.4. Malayalam

The Malayalam dataset is the most extensive among the four, containing tokens categorized into 'Malayalam,' 'English,' 'Mixed,' 'Name,' 'Number,' 'Location,' and 'Sym' for sentence boundaries. Similar to the other datasets, the tokens are presented in Roman script for standardized processing. The inclusion of categories like 'Number' and 'Sym' broadens the dataset's applicability, supporting a wide range of natural language processing tasks, particularly in language identification.

2.5. Tamil

The Tamil dataset includes tokens categorized into classes consistent with those used in the Tulu and Kannada datasets. These tokens are also presented in Roman script to facilitate uniform processing. This dataset is intended to support language identification in Tamil-English code-mixed text, providing a structured resource for exploring language mixing patterns specific to Tamil.

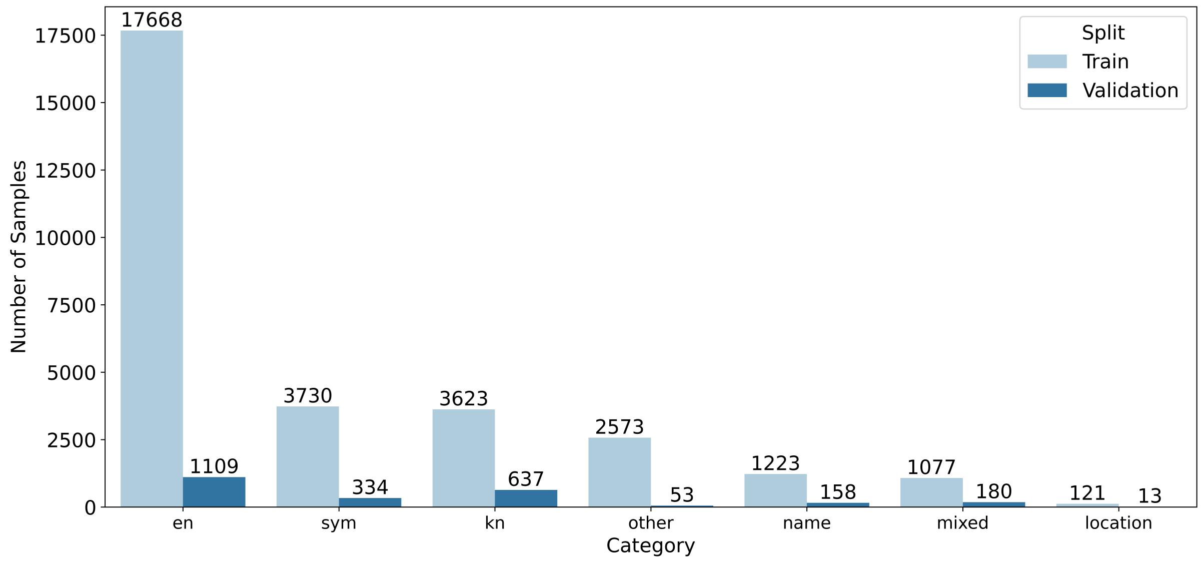

Figure 3: Train-val Target Distribution for Kannada language.

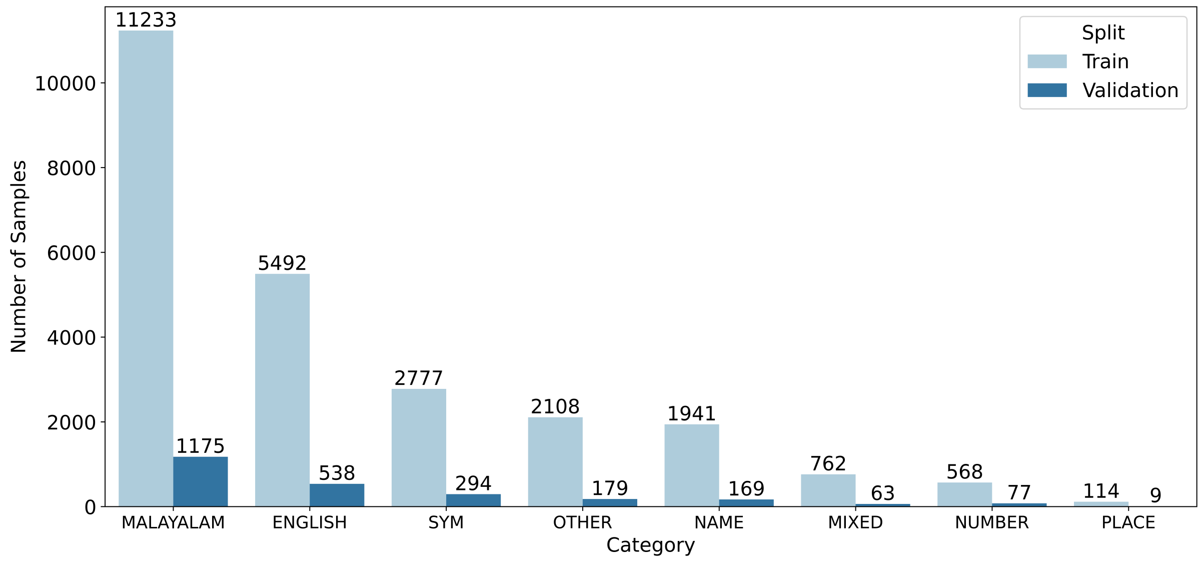

Figure 4: Train-val Target Distribution for Malayalam language.

3. Experimental Set-up

In this section, we discuss our approach and explain the experimental set-up details. We start with creating a validation strategy for each language. As the dataset for each languages are fairly unbalanced, we opt for Stratified K-Fold cross-validation with 5 folds. And while creating the splits, we set the random seed to 42 for Reproducibility.

3.1. Preprocessing

The preprocessing phase involved several cleaning and preparation tasks designed to optimize the datasets for feature extraction and modeling. Initially, the datasets were reviewed for missing values, irregularities, and non-textual noise. Any entries with missing values were removed, and non-standard symbols or excessive punctuation were stripped away to enhance data quality. Given the code-mixed

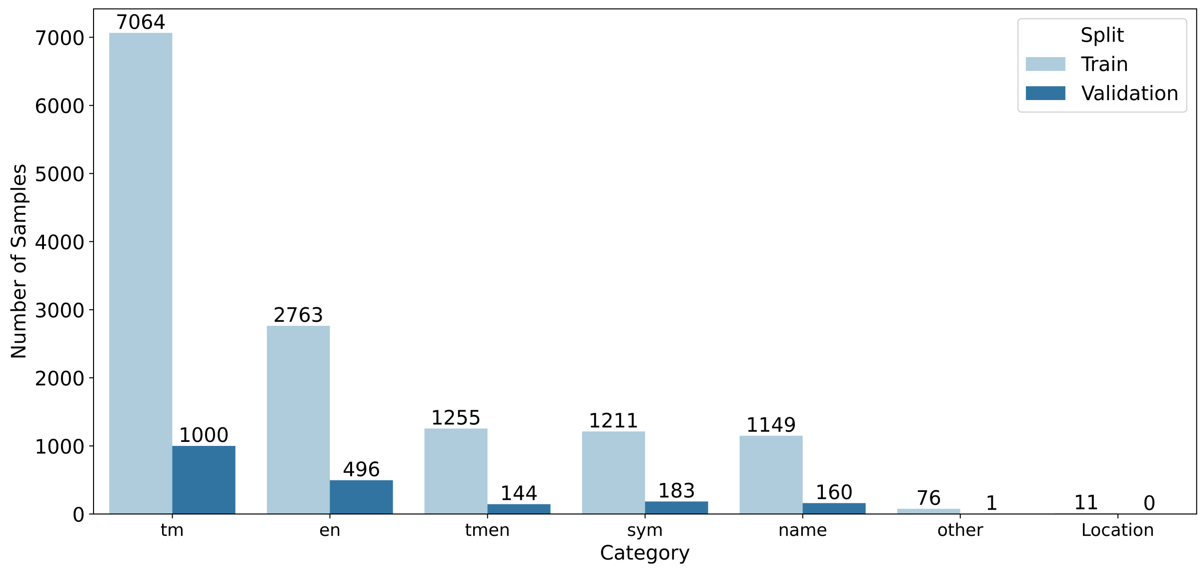

Figure 5: Train-val Target Distribution for Tamil language.

nature of the text, which included English interspersed with Tulu, Kannada, Tamil, or Malayalam, we employed language detection tools to tag and separate these instances, ensuring consistency across the datasets. To address the significant class imbalance, particularly for underrepresented languages like Tulu and Kannada, class weights were computed and applied during model training to mitigate bias towards the majority classes. Text data was then transformed into numerical form using various text embedding techniques, such as Count Vectorizer and TF-IDF with n-grams ranging from 1 to 4, we also used FastText[9] embeddings to capture both semantic and syntactic features. Target labels were encoded from categorical to numerical format using LabelEncoder, facilitating the training of machine learning models.

3.2. Modeling

The modeling phase involved a diverse set of machine learning algorithms to classify the text data for all four subtasks. We began with a suite of baseline models, including MultinomialNB, Logistic Regression, LinearSVC, KNeighborsClassifier, Decision Tree Classifier, Random Forest Classifier, SVC. These models were selected for their simplicity and effectiveness in handling text classification tasks, providing a solid foundation for initial experimentation and serving as benchmarks for further comparisons. As we aimed to improve performance, we incorporated more advanced ensemble models such as LGBMClassifier[10], XGBoostClassifier[11], and CatBoostClassifier[12], which are renowned for their high performance in structured data and text classification tasks. To fine-tune the models, we employed two hyperparameter optimization techniques: Grid Search and Optuna. Grid Search was used to systematically explore predefined hyperparameter grids for baseline models, including MultinomialNB, Logistic Regression, LinearSVC, KNeighborsClassifier, Decision Tree, and Random Forest. For more sophisticated models like Random Forest, LightGBM, XGBoost, and CatBoost, we utilized Optuna, a robust hyperparameter optimization framework that efficiently searches for optimal parameters using Bayesian optimization. Throughout the modeling process, all experiments, including data preprocessing, model training, and hyperparameter tuning, were consistently applied across the four subtasks (Tulu, Kannada, Tamil, and Malayalam) to ensure fair and meaningful performance comparisons among models and datasets. Each experiment explored different text embeddings and model configurations to determine the most effective strategies for each language classification task. During inference, models trained on each fold were ensembled by averaging the logits, which provided a comprehensive and robust classification output.

4. Results

Table 1
Macro-F1 Scores on Kannada Evaluation Data.

Models	CoVec(1,4)	Tf-Idf (1, 3)	Tf-Idf (1, 4)	FastText
Multinomial NB	0.784	0.789	0.813	-
Logistic Regression	0.853	0.835	0.853	0.240
SVC	0.856	0.844	0.852	0.807
K-Nearest Neighbour	0.815	0.810	0.802	0.748
Decision Tree	0.826	0.815	0.817	0.777
Random Forest	0.848	0.839	0.844	0.778
LightGBM	0.854	0.850	0.845	0.792
XgBoost	**0.860**	0.850	0.845	0.792
CatBoost	0.841	0.853	0.843	0.779

Table 2
Macro-F1 Scores on Tamil Evaluation Data.

Models	CoVec(1,4)	Tf-Idf (1, 3)	Tf-Idf (1, 4)	FastText
Multinomial NB	0.673	0.728	**0.732**	-
Logistic Regression	0.690	0.698	0.693	0.460
SVC	0.693	0.700	0.692	0.460
K-Nearest Neighbour	0.629	0.629	0.627	0.550
Decision Tree	0.679	0.647	0.658	0.515
Random Forest	0.669	0.680	0.677	0.597
LightGBM	0.684	0.681	0.680	0.546
XgBoost	0.697	0.683	0.677	0.645
CatBoost	0.703	0.635	0.682	0.537

Table 3
Macro-F1 Scores on Malayalam Evaluation Data.

Models	CoVec(1,4)	Tf-Idf (1, 3)	Tf-Idf (1, 4)	FastText
Multinomial NB	0.796	0.754	0.790	-
Logistic Regression	0.861	0.862	0.865	0.336
SVC	0.852	0.856	0.845	0.460
K-Nearest Neighbour	0.817	0.809	0.809	0.772
Decision Tree	0.833	0.825	0.835	0.754
Random Forest	0.818	0.860	0.850	0.755
LightGBM	**0.864**	0.846	0.858	0.766
XgBoost	0.858	0.849	0.860	0.769
CatBoost	0.832	0.834	0.839	0.756

The results from the CoLI Dravidian tasks, presented in tables 1, 2, 3, 4, demonstrate the effectiveness of various machine learning algorithms in language identification across Kannada, Tamil, Malayalam, and Tulu. XgBoost emerged as the leading model for Kannada, achieving the highest macro-F1 score of 0.860 with the Count Vectorizer ngram(1,4) Embedding set, highlighting its robustness in identifying language patterns. In Tamil, the Multinomial Naive Bayes model, using the Tf-Idf ngram(1,4) configuration, slightly outperformed other models with a macro-F1 score of 0.732, showcasing its efficiency in handling language-specific nuances. For Malayalam, LightGBM excelled, attaining a macro-F1 score of 0.864 with the Count Vectorizer ngram(1,4) representation. Logistic Regression was the top performer for Tulu, with a macro-F1 score of 0.847 using Tf-Idf ngram(1,4).

Table 4
Macro-F1 Scores on Tulu Evaluation Data.

Models	CoVec(1,4)	Tf-Idf (1, 3)	Tf-Idf (1, 4)	FastText
Multinomial NB	0.751	0.744	0.785	-
Logistic Regression	0.839	0.832	**0.847**	0.260
SVC	0.834	0.835	0.834	0.426
K-Nearest Neighbour	0.804	0.803	0.803	0.731
Decision Tree	0.801	0.802	0.791	0.715
Random Forest	0.831	0.835	0.834	0.748
LightGBM	0.833	0.817	0.819	0.756
XgBoost	0.831	0.823	0.825	0.749
CatBoost	0.825	0.803	0.812	0.742

Table 5
Leaderboard macro-f1 scores in Test Set

Language	Model	Score
Tamil	XgBoost + CoVec(1,4)	**0.730**
Kannada	SVC + Tf-Idf(1,3)	**0.857**
Malayalam	SVC + CoVec(1,4)	**0.868**
Tulu	SVC + Tf-Idf(1,3)	**0.838**

These findings are further corroborated by the leaderboard scores (refer to table 5), where we find that SVC model generally performs well across languages. The obtained results help us climb to 2nd/10 for Tamil, 4th/10 for Kanadda, 3rd/10 for Malayalam, and 3rd/9 for Tulu. The overall rankings highlight the strengths of different machine learning models and feature-engineered representations in multilingual language identification tasks

5. Conclusion

This study explored various machine learning models and text embeddings for word-level language identification in code-mixed texts involving Dravidian languages as part of the CoLI-Dravidian Shared Task@FIRE2024. We experimented with a wide range of models, from classical machine learning algorithms such as Logistic Regression, Support Vector Machines (SVM), Multinomial Naive Bayes, K-Nearest Neighbors (KNN), Decision Trees, and Random Forests, to advanced ensemble methods like XGBoost, CatBoost, and LightGBM. These models were trained using different embeddings, including Count Vectorizer and TF-IDF with n-gram ranges (1,3) and (1,4), as well as FastText embeddings. We also conducted extensive hyperparameter tuning to optimize each model's performance. Our results demonstrated that advanced models like XGBoost and LightGBM generally outperformed classical methods across most tasks, yet simpler models such as Logistic Regression also exhibited competitive performance when paired with optimized text embeddings and hyperparameter tuning. The overall effectiveness of our methodology was reflected in our rankings, where we secured Rank 4 in Kannada, Rank 3 in Malayalam, Rank 3 in Tulu, and Rank 2 in Tamil. These results underscore the complexity and variability of code-mixed language processing, highlighting the necessity for more tailored approaches depending on the linguistic context.

Moving forward, we aim to extend our work to include deep learning models and transformer-based architectures, explore zero-shot and few-shot learning techniques, and develop a unified model capable of handling multiple languages more effectively. These steps will further enhance the robustness and inclusivity of NLP applications for under-resourced languages.

Declaration on Generative AI

During the preparation of this work, the author(s) used Chat GPT-4 in order to:Grammar and spelling check. Using this tool, the author(s) reviewed and edited the content as needed and take full responsibility for the publication's control.

References

[1] A. Hande, S. U. Hegde, B. R. Chakravarthi, Multi-task learning in under-resourced dravidian languages, Journal of Data, Information and Management 4 (2022) 137–165. URL: https://doi.org/10.1007/s42488-022-00070-w. doi:10.1007/s42488-022-00070-w.

[2] B. R. Chakravarthi, R. Priyadharshini, V. Muralidaran, N. Jose, S. Suryawanshi, E. Sherly, J. P. McCrae, Dravidiancodemix: sentiment analysis and offensive language identification dataset for dravidian languages in code-mixed text, Language Resources and Evaluation 56 (2022) 765–806. URL: https://doi.org/10.1007/s10579-022-09583-7. doi:10.1007/s10579-022-09583-7.

[3] S. Jain, K. Agarwal, Usefulness of graphemes in word-level language identification in code-mixed text, in: J. P. Sahoo, A. K. Tripathy, M. Mohanty, K.-C. Li, A. K. Nayak (Eds.), Advances in Distributed Computing and Machine Learning, Springer Singapore, Singapore, 2022, pp. 174–185.

[4] N. Sarma, R. Sanasam Singh, D. Goswami, Switchnet: Learning to switch for word-level language identification in code-mixed social media text, Natural Language Engineering 28 (2022) 337–359. URL: https://www.cambridge.org/core/product/5E8F2A0046A559107F025E7B9DEE155B. doi:10.1017/S1351324921000115.

[5] A. Hegde, F. Balouchzahi, S. Butt, S. Coelho, K. G, H. S Kumar, S. D, S. Hosahalli Lakshmaiah, A. Agrawal, Overview of CoLI-Dravidian: Word-level Code-mixed Language Identification in Dravidian Languages, in: Forum for Information Retrieval Evaluation FIRE - 2024, 2024.

[6] F. Balouchzahi, S. Butt, A. Hegde, N. Ashraf, S. Hosahalli Lakshmaiah, G. Sidorov, A. Gelbukh, Overview of CoLI-Kanglish: Word Level Language Identification in Code-mixed Kannada-English Texts at ICON 2022, in: 19th International Conference on Natural Language Processing Proceedings, 2022.

[7] A. Hegde, M. D. Anusha, S. Coelho, H. L. Shashirekha, B. R. Chakravarthi, Corpus creation for sentiment analysis in code-mixed Tulu text, in: M. Melero, S. Sakti, C. Soria (Eds.), Proceedings of the 1st Annual Meeting of the ELRA/ISCA Special Interest Group on Under-Resourced Languages, European Language Resources Association, Marseille, France, 2022, pp. 33–40. URL: https://aclanthology.org/2022.sigul-1.5.

[8] H. L. Shashirekha, F. Balouchzahi, M. D. Anusha, G. Sidorov, Coli-machine learning approaches for code-mixed language identification at the word level in kannada-english texts, 2022. URL: https://arxiv.org/abs/2211.09847. arXiv:2211.09847.

[9] P. Bojanowski, E. Grave, A. Joulin, T. Mikolov, Enriching word vectors with subword information, 2017. URL: https://arxiv.org/abs/1607.04606. arXiv:1607.04606.

[10] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, T.-Y. Liu, Lightgbm: A highly efficient gradient boosting decision tree, in: I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (Eds.), Advances in Neural Information Processing Systems, volume 30, Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper_files/paper/2017/file/6449f44a102fde848669bdd9eb6b76fa-Paper.pdf.

[11] T. Chen, C. Guestrin, Xgboost: A scalable tree boosting system, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2016. URL: http://dx.doi.org/10.1145/2939672.2939785. doi:10.1145/2939672.2939785.

[12] L. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, A. Gulin, Catboost: unbiased boosting with categorical features, in: S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, R. Garnett (Eds.), Advances in Neural Information Processing Systems, volume 31,

Curran Associates, Inc., 2018. URL: https://proceedings.neurips.cc/paper_files/paper/2018/file/14491b756b3a51daac41c24863285549-Paper.pdf.