Fusion-based approach for sentiment Analysis of YouTube comments in Tamil code-mixed data

DIKSHITHA VANI V[*], DEVISRI S R[*], B. BHARATHI, and D. THENMOZHI[*], Sri Sivasubramaniya Nadar College of Engineering, India

This paper discusses the importance of sentiment analysis on code-mixed texts in the Dravidian language (Tamil-English), particularly on social media. Code-mixing is a common practice among multilingual communities, which makes the task of sentiment analysis challenging due to the complex linguistic levels involved. Forum for Information Retrieval and Evaluation(FIRE2022) presents a new gold standard corpus for sentiment detection of code-mixed text in Dravidian languages (Tamil-English, Malayalam-English, and Kannada-English) in their shared Task - A. For this proposed work, we have considered the Tamil-English code mixed data. In this work, standalone algorithms like extra trees, MLP classifiers, and SVM are used, and also focus on discussing the effect of feature fusion and decision fusion models on the Tamil code-mixed data. The results of this study provide insights into the applicability of standard algorithms for sentiment analysis of code-mixed texts and can aid in developing efficient systems for sentiment analysis on social media in Dravidian languages.

Additional Key Words and Phrases: MLP, SVM, extra trees, feature fusion, decision fusion

ACM Reference Format:
Dikshitha Vani V, Devisri S R, B. Bharathi, and D. Thenmozhi. 2023. Fusion-based approach for sentiment Analysis of YouTube comments in Tamil code-mixed data. 1, 1 (March 2023), 9 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

In recent years, social media platforms such as YouTube have become an integral part of our daily lives, providing a space for users to upload, share and view videos. With over two billion monthly active users, YouTube has become one of the largest social media platforms in the world to express their opinions and engage in conversations with others. This results in large amount of data which can be used for sentiment analysis.

In this paper, we present a sentiment analysis for YouTube comments in code-mixed Dravidian language. Code-mixing refers to mixing of two or more languages in a single text or conversation. In case of code-mixed Dravidian language, Dravidian languages are combined with other languages such as English or Hindi. This paper focuses on analysing the comments related to the Tamil language code-mixed with English.

Tamil is a Dravidian language natively spoken by the Tamil people of South Asia. Tamil is the official language of the Indian state of Tamil Nadu, the sovereign nations of Sri Lanka and Singapore, and the Indian Union territory of

[*] All authors contributed equally to this research.

Authors' address: Dikshitha Vani V, dikshithavani2010541@ssn.edu.in; Devisri S R, devisri2010569@ssn.edu.in; B. Bharathi, bharathib@ssn.edu.in; D. Thenmozhi, theni_d@ssn.edu.in, Sri Sivasubramaniya Nadar College of Engineering, Chennai, Tamil Nadu, India.

Puducherry. Tamil is also spoken by significant minorities in the four other South Indian states of Kerala, Karnataka, Andhra Pradesh, and Telangana, and the Union Territory of the Andaman and Nicobar Islands. It is also spoken by the Tamil diaspora found in many countries, including Malaysia, Myanmar, South Africa, the United Kingdom, the United States, Canada, Australia, and Mauritius. Tamil is also natively spoken by Sri Lankan Moors. One of 22 scheduled languages in the Constitution of India, Tamil was the first to be classified as a classical language of India.

In this paper, we aim to perform the classification of sentiments from the Tamil code-mixed data using standalone models like Multilayer Layer Perceptraon (MLP), extra Trees and Support Vector Machine (SVM) classifier. Further, the paper focuses on decision fusion models with Extra trees, MLP and SVM classifier models. The paper also discusses feature fusion where count vectorizer, tf-idf and bert-base-nli-mean-tokens are used for feature extraction on ExtraTrees, SVM and MLP classifer models.

The rest of the paper is organized as follows-Section 2 describes other related work on Code-Mixed language models. The dataset for the shared task used for this task are described in Section 3. Section 4 discusses the standalone classifier models used in this paper. Section 5 discusses fusion models. Results and error analysis are presented in Section 6. Section 7 concludes the paper.

2 RELATED WORKS

Code-mixed data presents unique challenges for sentiment analysis due to the mixing of languages, diverse grammatical rules, and cultural nuances. Researchers and practitioners have been actively exploring various techniques to accurately analyze the sentiment of code-mixed data, with the goal of developing robust sentiment analysis models that can better understand the intricacies of multilingual communication.

The author [1] uses three different feature sets namely char,char sequence and syllables to train three different Machine Learning Models: Linear Support Vector Machine(LSVM), Linear Regression(LR) and Multi Layer Perceptron(MLP) and the majority voting of all the predictions of the clasifiers is used to classify a given sentiment.

[2] proposes two hybrid models with Convolutional Neural Network (CNN) and Bidirectional Long-Short-Term-Memory (Bi-LSTM) . The first model comprises of two parallel CNN networks to extract at character level and word level(CNN (c) + CNN (w)).The second model comprises of CNN and Bi-LSTM for extraction at character and word level respectively.(CNN (c) + Bi-LSTM (w)).

The lack of sufficient resources for most of the world's languages has been a classic problem in NLP, resulting in hugely imbalanced progress in the development of language technologies for different languages. One of the ways of handling this problem is to make use of multilingual and transfer learning methods for training the systems. The author [3] uses pre-trained models such as mBERT(multilingual BERT) and XLM-R to test this method.

The authors [7] propose a method for both sentiment analysis and offensive language identification for code-mixed data using machine learning, deep learning, and pre-trained models like BERT, RoBERTa, and adapter-BERT. The author also uses BERT and GLOVE embeddings which work better for pre-trained models. The adapter-BERT model gives better results with an accuracy of 65 percent for sentiment analysis and 79 percent for offensive language identification.

The author [6] compares the performance of the proposed traditional machine learning, deep learning, transfer learning and hybrid deep learning models. For feature extraction TF-IDF method is used. The classification is done using four traditional machine learning models (RF, LSVC, MNB and LR), four deep learning models (LSTM, BiLSTM, BiGRU and CNN), one transfer learning technique IndicBERT and four hybrid models (CNN+LSTM, LSTM+CNN, CNN+BiLSTM and BiLSTM+CNN). The models are evaluated based on precision, recall, F1-score, accuracy, macro-average, weighted average, and confusion matrix. The hybrid deep learning model, especially the CNN+BiLSTM model performs better, with an accuracy of 0.66.

The author [5] uses a multi-language pre-training model called XLM-RoBERTa, which not only inherits the XLM training method but also draws on the ideas of RoBERTa. To improve the overall classification performance of the model, K-fold ensemble method is used. The evaluation method for the model is F Score.

The author [8] uses three kinds of classic systems - an SVM classifier, a logistic classifier, and a Perceptron model. For feature extraction, n CountVectorizer is used.[9] comes up with a solution to analyze sentiments for class imbalanced code-mixed data using a sampling technique combined with Levenshtein distance metrics. Furthermore, this paper compares the performances of various machine learning approaches namely, Random Forest Classifier, Logistic Regression, XGBoost classifier, Support Vector Machine, and Naïve Bayes Classifier using F1- Score.

Supervised learning methods like Naïve Bayes (NB) Classifier, Support Vector Machine (SVM) and Maximum Entropy (ME) to classify data into positive or negative categories in [4].

3 PROPOSED METHODOLOGY

This section focuses on exploration of dataset, feature extraction and training the models such as extra trees, MLP and SVM. Early fusion such as feature level fusion and late fusion such as decision level fusions are explored in this section.

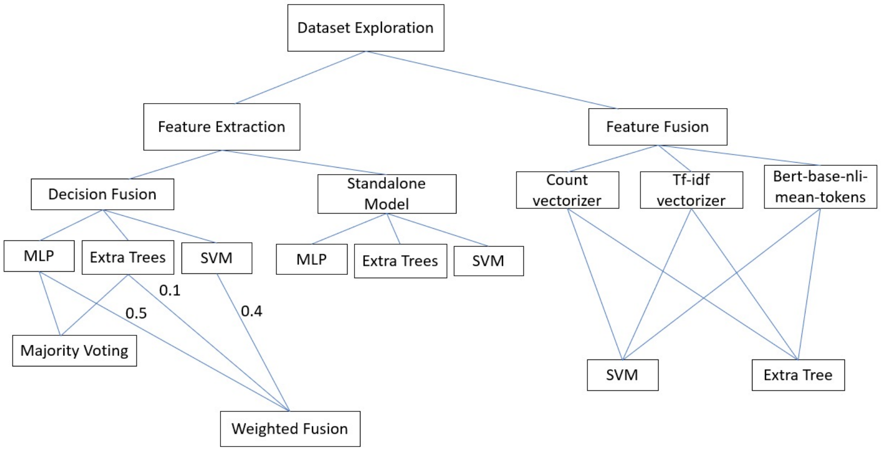


Fig. 1. Overall Process Flow

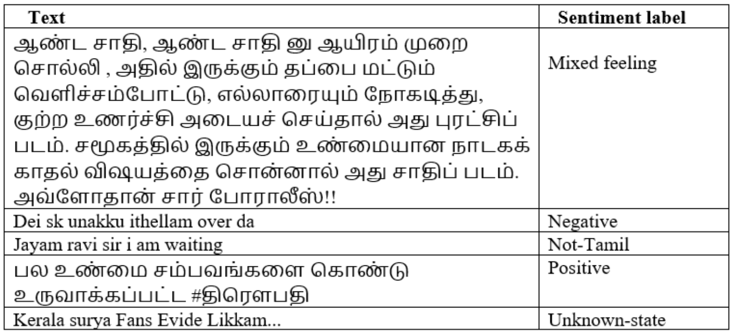
3.1 Data Set exploration

The data set given by the shared task organizers contains a training set consisting of 35657 instances and a development set consisting of 3963 instances. The training and development set contains the text(comment) and category fields. The text is classified into any of the five categories- Positive, Negative, Mixed_feelings, unknown_state, and not-Tamil, the number of samples in each category in shown in Table 1.

Label	No. of training samples	No. of development samples
Positive	20070	2257
Negative	4271	480
Mixed feelings	4020	438
Unknown state	20070	611
Not Tamil	1667	176

Table 1. Statistics of data distribution for Tamil code-mixed data

In the proposed work, we have used the training set to learn the model and the development data set to test the model. The sample text for each of the sentiment classes are shown in fig. 2.



Text	Sentiment label
ஆண்ட சாதி, ஆண்ட சாதி னு ஆயிரம் முறை சொல்லி , அதில் இருக்கும் தப்பை மட்டும் வெளிச்சம்போட்டு, எல்லாரையும் நோகடித்து, குற்ற உணர்ச்சி அடையச் செய்தால் அது புரட்சிப் படம். சமூகத்தில் இருக்கும் உண்மையான நாடகக் காதல் விஷயத்தை சொன்னால் அது சாதிப் படம். அவ்ளோதான் சார் போராலீஸ்!!	Mixed feeling
Dei sk unakku ithellam over da	Negative
Jayam ravi sir i am waiting	Not-Tamil
பல உண்மை சம்பவங்களை கொண்டு உருவாக்கப்பட்ட #திரௌபதி	Positive
Kerala surya Fans Evide Likkam...	Unknown-state

Fig. 2. Example Tamil code-mixed data for each sentiment class

3.2 Classifier Models

This section lists the models used in this paper each of which is explained in a separate section.

(1) Standalone Models
(2) Fused Models
  (a) Decision Fusion
  (b) Feature Fusion

4 STANDALONE MODELS

In this paper, we have used Extra Trees Classifier, MLP, and SVM classifier models.

4.1 Extra Trees classifier

Extra Trees is a type of ensemble learning model that belongs to the family of decision tree algorithms. Extra Trees creates an ensemble of decision trees, each of which is trained on a random subset of the training data and a random subset of the features. The trees in the ensemble vote on the class label for each test instance, and the class with the most votes is assigned as the predicted label. The sklearn.ensemble package in python is used to import the ExtraTreesClassifier model.

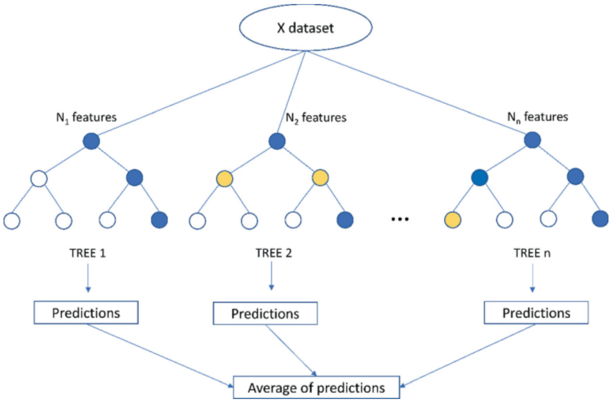


Fig. 3. Extra Trees classifier model

The hyperparameters used for MLP classifier are n_estimator and random_state. The n_estimators hyperparameter determines the number of decision trees that will be created in the ExtraTrees model. Increasing the value of n_estimators can improve the accuracy of the model. The random_state hyperparameter is used to control the randomness in the ExtraTrees model. Setting a fixed value for random_state ensures that the model will be reproducible, as the same sequence of random numbers will be used to train the model each time it is run.

Count vectorizer was for feature extraction. The training data is used to fit into the extra trees classifier model and the development set is used to predict the labels. A weighted average F1-score of 0.54 was observed.

4.2 MLP classifier

MLP, or Multi-Layer Perceptron is a feedforward neural network. The Information flows in one direction, from the input layer to the output layer, through one or more hidden layers. Each layer consists of a set of neurons, which perform computations on the input data. The neurons in the input layer receive the input data, and the neurons in the output layer produce the output predictions. The neurons in the hidden layers perform computations on the input data and gradually learn to identify complex patterns in the data. MLP classifier is imported from sklearn.neural_network package in python.

The hyperparameters used for MLP classifier are hidden_layer_sizes and max_iter. The hidden_layer_sizes is used to specify the number of layers and the number of nodes we want to have in the network. Each element in the tuple represents the number of nodes at the ith position where i is the index of the tuple. Thus the length of the tuple denotes the total number of hidden layers in the network. The number of hidden layers in the MLP can greatly affect its performance. Adding more hidden layers may allow the MLP to model more complex relationships. max_iter denotes the number of epochs the training algorithm will run for.

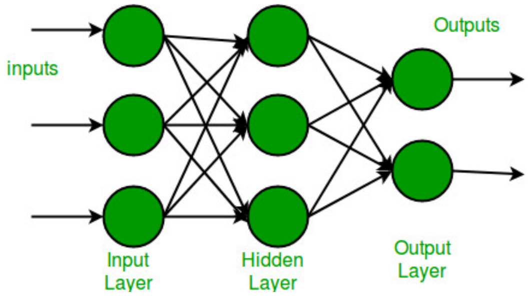


Fig. 4. Multilayer perceptron model

Count vectorizer was for feature extraction. The training data is used to fit into the MLP classifier model and the development set is used to predict the labels. A weighted average F1-score of 0.58 was observed.

4.3 SVM classifier

It is a discriminative classifier that works by finding the hyperplane in a high-dimensional space that maximally separates the data points of different classes.

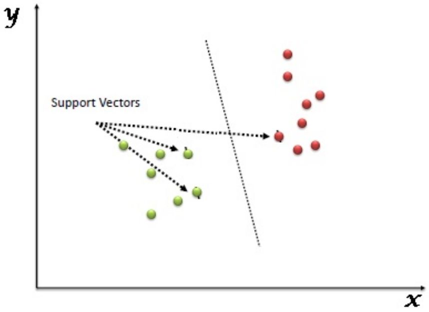


Fig. 5. SVM classifier model

The hyperparameters used in SVM classifier are kernal and C-value. The kernel hyperparameter determines the type of function used to transform the input data into a higher-dimensional space, where it can be more easily separated by a hyperplane. The type of kernel used in the model is the linear kernel which produces a linear decision boundary. The C hyperparameter controls the trade-off between maximizing the margin and minimizing the classification error. A larger value of C will result in a narrower margin and a smaller number of misclassified samples, while a smaller value of C will result in a wider margin and a larger number of misclassified samples.

Count vectorizer was for feature extraction. The training data is used to fit into the SVM classifier model and the development set is used to predict the labels. A weighted average F1-score of 0.58 was observed.

The scores for the standalone models are shown in table 2.

Table 2. Results of Standalone Models

Classifier Model	Precision	Recall	F1 Score
ExtraTress Classifier	0.61	0.62	0.54
MLP classifier	0.57	0.59	0.58
SVM classifier	0.58	0.63	0.58

5 FUSION APPROACHES

Fused models in NLP are models that combine multiple machine learning algorithms or models to improve performance on a specific task. Fused models can take many different forms, including ensembles of models, models that use multiple data sources or features, or models that combine different types of machine learning techniques.

Fusing multiple models can help overcome the limitations of individual models and improve performance by leveraging the strengths of each model.

5.1 Decision Fusion

Decision fusion models are a type of fused model in which multiple classifiers are combined to improve performance on a specific task. Decision fusion models operate by aggregating the decisions made by individual classifiers into a single decision, often through a voting or weighting scheme which is shown in Fig.6.

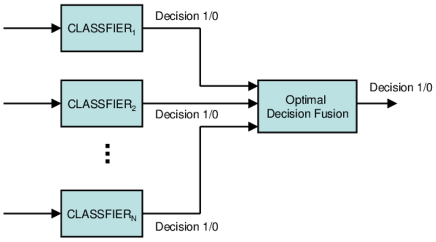


Fig. 6. Decision Fusion approach

5.1.1 Majority Voting. In this approach, each classifier casts a vote for its predicted label, and the label with the most votes is chosen as the final prediction. In our work, we used Extra Trees and MLP classifier models and performed a majority voting.

The F1-score of the fused model presents a better score than either of the models with a F1-score of 0.59.

Table 3. Decision Fusion Model Results

Decision Fusion	**Precision**	**Recall**	**F1 Score**
Majority Voting: ExtraTress+MLP	0.58	0.60	**0.59**
Weighted decision: ExtraTrees+MLP+SVM	0.58	0.57	0.56

5.1.2 Weighted Fusion. In this approach, each classifier's vote is weighted according to its confidence or accuracy, and the final prediction is made based on the weighted votes.In our work, we used Extra Trees, MLP and SVM classifier models and performed a weighted fusion.

The Extra tree classifier was given a weight of 0.1, MLP was given a weight of 0.5 and SVM was given a weight of 0.4 and it produced an F1 score of 0.56 which is greater than that of Extra tree classifier model.
The results of the Fusion models are depicted in table 3.

5.2 Feature Fusion

Feature fusion refers to the process of combining multiple representations or features from different sources or modalities into a single, more informative representation. In machine learning and computer vision, feature fusion can be used to improve the accuracy and robustness of models by incorporating complementary information from multiple sources.

Table 4. Performance of fusion of features in the proposed approach

Classifier model	Precision	Recall	F1-score
Extra Tree	0.59	0.62	0.53
SVM	0.67	0.59	0.45

Count-vectorizer, TF-IDF vectorizer and bert-base-nli-mean-tokens were used for feature extraction. Applying majority voting in feature fusion to classifier models - Extra tree and SVM produces a F1- score of 0.53 and 0.45 respectively.

6 RESULTS AND ERROR ANALYSIS

In this section, we present the evaluation of our model and submitted results for sentiment analysis in Tamil code-mixed data The performance of our proposed models is examined using evaluation measures such as precision, recall and F1-score. From the tables tables 2,3, and 4, it has been noted that Majority voting between Extra Trees and MLP has the highest F1- score. From the tables 2,3, and 4, another important point noted was that recall is better than precision in majority of the proposed approaches. For better system false positives should be avoided that is achieved in the proposed results. Another reason for misclassification was that the category of unknown state is high in the training data. The unkown state category may belong to the other sentiment class which was not known during training.

7 CONCLUSION

Sentiment analysis is a rapidly growing field in natural language processing that aims to extract and classify the underlying sentiment of a piece of text. With the increasing use of code-mixing in online communication, where individuals switch between two or more languages within a single conversation or sentence, the need for sentiment analysis of code-mixed data has become increasingly important.

In this research paper, we have presented an approach to building standalone models for the code-mixed Dravidian Tamil-English dataset using Extratree, SVM, and MLP models. We have also explored decision and feature fusion techniques to improve the overall performance of the models. These are some of the ways in improving the performance for a code-mixed data set. The experimental findings demonstrate that fusion models outperform different baseline models for stance detection. We can increase the efficiency of the suggested modes by using context-aware domain-specific embeddings.

REFERENCES

[1] Fazlourrahman Balouchzahi, Hosahalli Lakshmaiah Shashirekha, and Grigori Sidorov. Cosad-code-mixed sentiments analysis for dravidian languages. In *CEUR Workshop Proceedings*, volume 3159, pages 887–898. CEUR-WS, 2021.

[2] Abhinav Kumar, Sunil Saumya, and Jyoti Prakash Singh. Nitp-ai-nlp@ dravidian-codemix-fire2020: A hybrid cnn and bi-lstm network for sentiment analysis of dravidian code-mixed social media posts. In *FIRE (Working Notes)*, pages 582–590, 2020.

[3] Ritesh Kumar, Bornini Lahiri, Atul Kr Ojha, and Akanksha Bansal. Comma@ fire 2020: Exploring multilingual joint training across different classification tasks. In *FIRE (Working Notes)*, pages 823–828, 2020.

[4] Nurul Husna Mahadzir et al. Sentiment analysis of code-mixed text: a review. *Turkish Journal of Computer and Mathematics Education (TURCOMAT)*, 12(3):2469–2478, 2021.

[5] Xiaozhi Ou and Hongling Li. Ynu@ dravidian-codemix-fire2020: Xlm-roberta for multi-language sentiment analysis. In *FIRE (Working Notes)*, pages 560–565, 2020.

[6] Kogilavani Shanmugavadivel, Sai Haritha Sampath, Pramod Nandhakumar, Prasath Mahalingam, Malliga Subramanian, Prasanna Kumar Kumaresan, and Ruba Priyadharshini. An analysis of machine learning models for sentiment analysis of tamil code-mixed data. *Computer Speech & Language*, 76:101407, 2022.

[7] Kogilavani Shanmugavadivel, VE Sathishkumar, Sandhiya Raja, T Bheema Lingaiah, S Neelakandan, and Malliga Subramanian. Deep learning based sentiment analysis and offensive language identification on multilingual code-mixed data. *Scientific Reports*, 12(1):21557, 2022.

[8] Deepesh Sharma. Tads@ dravidian-codemix-fire2020: Sentiment analysis on codemix dravidian language. In *FIRE (Working Notes)*, pages 615–619, 2020.

[9] R Srinivasan and CN Subalalitha. Sentimental analysis from imbalanced code-mixed data using machine learning approaches. *Distributed and Parallel Databases*, pages 1–16, 2021.