

ARTICLE

Sentiment analysis of code-mixed Dravidian languages leveraging pretrained model and word-level language tag

Supriya Chanda¹ , Anshika Mishra² and Sukomal Pal¹

¹Department of Computer Science and Engineering, Indian Institute of Technology (BHU), Varanasi, Uttar Pradesh, India and ²Vellore Institute of Technology Bhopal, Bhopal, Madhya Pradesh, India

Corresponding author: Supriya Chanda; Email: supriyachanda.rs.cse18@itbhu.ac.in

(Received 25 December 2022; revised 20 July 2023; accepted 25 October 2023; first published online 11 September 2024)

Special Issue on ‘Natural Language Processing Applications for Low-Resource Languages’

Abstract

The exponential growth of social media data in the era of Web 2.0 has necessitated advanced techniques for sentiment analysis. While sentiment analysis in monolingual datasets has received significant attention that in code-mixed datasets still need to be studied more. Code-mixed data often contain a mixture of monolingual content (might be in transliterated form), single-script but multilingual content, and multi-script multilingual content. This paper explores the issue from three important angles. What will be the best strategy to deal with the data for sentiment detection? Whether to train the classifier with the whole of the dataset or only with the pure code-mixed subset from the dataset? How much important is the language identification (LID) for the task? If LID is to be done, how, and when will it be used to yield the best performance? We explore the questions in the light of three datasets of Tamil–English, Kannada–English, and Malayalam–English YouTube social media comments. Our solution incorporated mBERT and an optional LID module. We report our results using a set of metrics like precision, recall, F_1 score, and accuracy. The solutions provide considerable performance gain and some interesting insights for sentiment analysis from code-mixed data.

Keywords: text classification; sentiment analysis; code-mixing; Dravidian languages; mBERT

1. Introduction

Sentiment analysis or opinion mining is a field in natural language processing (NLP), which involves identifying and extracting subjective information from a text snippet. This is useful in various applications. For example, from customer feedback data, companies can use sentiment analysis to understand how customers feel about their products or services and identify areas for improvement (Pang and Lee 2004). In market research, sentiment analysis can help businesses track the opinions of customers or competitors in real time and make informed decisions on marketing and product development. With social media monitoring, organizations can monitor the sentiment of their brand on social media platforms and fine tune their responses to negative or positive comments/feedback and calibrate their positions in the market and society. Governments and media organizations also can use sentiment analysis to track public mood on a specific issue or a candidate. Overall, sentiment analysis is crucial because it helps organizations understand how people feel about specific topics or products, which enables them to make informed business decisions and improve customer satisfaction.

With the advent of Web 2.0, social media sites have become a part of people's daily routine. The number of social media sites and their active users are continuously on the rise generating a humongous volume of text data including feelings and views. The popularity of a post or video in social media platforms or that of a product in an online retailing website nowadays largely depends on users' or customers' sentiments, reviews, and collective ratings (Thavareesan and Mahesan 2019). For instance, positive feedback (Thavareesan and Mahesan 2020a) could contribute to a post's/product's increased popularity (Thavareesan and Mahesan 2020b). Contrarily, unfavorable or negative opinions could lead to the rejection or disposal of a particular post/product. While social media posts help a user decide whether to watch a movie or purchase a product, the sentiments gleaned from the posts like customer reviews, often provide helpful insights to different manufacturers, online retailers, and content creators (movie producers, automakers, etc.) to strengthen their areas of weakness and leverage their areas of strength (Balouchzahi and Shashirekha 2021). A case in point, the adverse comments/opinions expressed by users over WhatsApp's new policy on sharing of data, location, and business messages have significantly decreased the platform's popularity (Hern 2021). Sentiment analysis (SA), the process of automatically assessing these user-posted reviews or feelings to ensure correct sentiment identification and categorization, is thus becoming increasingly important nowadays.

In the early days of social media (SM), only the most affluent people had the privilege of using it as the internet was not that easily accessible to all. Consequently, majority of the comments, product feedback, and opinions expressed on the social networking sites were from educated and affluent in the society. The posts were thus mostly monolingual in nature: like English, French, German, Chinese, Japanese, and Korean. With the gradual advancement of technology, smartphones with rich GUIs, and multilingual keypads, users began writing in their own native languages and scripts along with Roman scripts (Ortiz-Ospina 2019). The increase in volume of the user-generated texts was matched by the efforts to investigate and understand different aspects expressed in the text. SA was certainly one of them initially to assess the various attitudes expressed in languages with rich resources, such as English and Spanish. But later on, when people belonging to a plethora of cultural/linguistic backgrounds got access to SM, the nature of comments, product reviews, and opinions gradually saw a evolution. Slowly but surely more and more users began to make content in informal settings on social networking websites and product review boards. Additionally, in order to enhance the user experience, these platforms made sure that the user could express their thoughts in a way that made them feel comfortable, that is, in their native language or switching between one or more languages throughout a single conversation. All this together contributed to "user-generated" content. The prevailing landscape of NLP systems predominantly revolves around training models on formal languages characterized by proper grammar rules. Hence, SA systems primarily focused on monolingual data pertaining to high-resource languages. However, a significant challenge arose to deal with "user-generated" content in under-resourced settings, where such content is often intertwined with English or other high-resource languages. Globally, around 3.5 percent of all tweets are code-switched (Rijhwani *et al.* 2017), while 17.2 percent of posts and comments on Indian SM are code-mixed (Bali *et al.* 2014). This integration of languages gives rise to numerous complications, rendering conventional systems incapable of effectively processing code-mixed "user-generated" content.

In such scenarios, the limitations of existing systems become evident, as they struggle to accommodate the intricacies and complexities associated with code-mixed language usage. The prevalence of code-mixed content necessitates the development of robust and adaptable sentiment analysis systems capable of handling this unique linguistic phenomenon. Addressing this gap in research is crucial for enabling effective sentiment analysis in diverse linguistic contexts, particularly those characterized by code-mixed user-generated content.

The simultaneous usage of two or more languages in a document, paragraph, comment, sentence, phrase, or on a word level is known as "code-mixing" or "code-switching." It is a unique characteristic used for communication in bilingual/multilingual communities

(Barman *et al.* 2014) driven by discursive, pragmatic, sociolinguistic, and structural factors (Sridhar 1978). Because code-mixing is a widespread verbal phenomenon in all conversations (voice and text-based interactions) between multilingual speakers, most SM comments are code-mixed in a multilingual society. Code-mixing can thus be defined as the process by which a bilingual or multilingual speaker alters an utterance in a different language. In terms of code-mixing activities, the large majority of language pairings are under-resourced. For example, in India, due to technological limitations and difficulties in adopting Indian language scripts, users typically utilize a combination of Roman letters along with English words in lieu of their native scripts. Writing text using a non-native script is known as transliteration. A large portion of code-mixed data contains transliteration. The intricacy of evaluating texts with code-mixing lies at several levels, such as words, phrases, and sentences. This, coupled with sentiments or reviews published in different languages makes SA a challenging task. Moreover, as there are no guidelines for writing in code-mixed texts, they frequently include inaccurate and incomplete phrases, non-standard abbreviations, and words with repeated letters, e.g. aaaaaa, gdn8, helloooooo, gooooooooood, soooooorrryyy, etc. Because traditional SA models are unable to capture the meaning of the sentences in code-mixed text, these complications lead to the development of novel SA models using multilingual word embedding.

Over the years, work done on code-mixed language was quite minimal, with research focusing mostly on Hindi–English, Spanish–English, and Bengali–English. However, in recent years, most academic institutions and industries have teamed together to work on a number of projects containing code-mixed data. In the present work, we have attempted to investigate sentiment analysis for Dravidian languages. Dravidian languages are most prevalent in the southern part of India (Chakravarthi *et al.* 2020). This language family is composed of the four major literary languages, Tamil (ISO 639-3: tam), Telugu (ISO 639-3: tel), Malayalam (ISO 639-3: mal), and Kannada (ISO 639-3: kan). These four languages are recognized by the Indian government among the 22 scheduled languages granting them official status. Apart from India, Singapore and Sri Lanka both recognize Tamil as an official language. Despite having millions of speakers, there are not enough tools and resources to construct reliable NLP applications in these languages. Each of the Dravidian languages has its own script and is agglutinative in nature. Initially, Dravidian languages were recorded using a Tamil script on cave walls in Tamil Nadu's Madurai and Tirunelveli regions during the second century BCE. Among the Dravidian languages, majority of the people speak Telugu, Tamil, Kannada, and Malayalam (in descending order of the number of speakers) (Chakravarthi, Priyadharshini, and Muralidaran 2022). Despite the fact that these languages have their own scripts, SM users frequently use the Roman script since it is simple to use and is available in majority of computers and mobile devices. In our present research we have used three Dravidian code-mixed languages viz, Tamil–English, Kannada–English, and Malayalam–English, which were part of the Dravidian CodeMix sentiment analysis task conducted at FIRE 2020 and 2021. The goal of the shared task was to determine the sentiment polarity of code-mixed data from Dravidian language pairs (Malayalam–English, Tamil–English, and Kannada–English) from the comment section of YouTube videos that were gathered via SM. The text was grouped into five categories: Positive, Negative, Mixed_feelings, unknown_state, and not-*<language>*.^a The shared task was scheduled for two years in a row. The task spawned a number of research issues, and the dataset was offered to investigate and explore them. In this work, we focus on the following research questions (RQs).

- *RQ-1*: When to call a dataset code-mixed? If a given dataset contains a mixture of pure monolingual, transliterated monolingual, single-script multilingual, and mixed script multilingual text, what is the best strategy for conducting sentiment analysis?

^aThe language might be Tamil, Kannada, or Malayalam

- RQ-2: How important is the language identification (LID) task for code-mixed data processing? Can a separate language identification model with a pretrained model improve the overall system performance? We seek to find the answer particularly, with respect to sentiment analysis task.
- RQ-3: Is it possible to see SA as a multilevel problem rather than a multi-class problem? If yes, can a multilevel hierarchical model enhance the performance?

The rest of the paper is organized as follows. In Section 2, we discuss the existing sentiment analysis approaches in other code-mixed languages. The dataset is described in Section 4 along with a description of the preprocessing approaches used. We detail the proposed methodology and experiment setup in Section 5. In Section 6, we present the results and compare the performance against the state-of-the-art model according to the standard evaluation metrics along with error analysis and a discussion. Finally, we conclude in Section 7.

2. Related work

This section summarizes earlier work carried out on sentiment analysis on different code-mixed languages. Sentiment analysis is an exclusive field of study, this is employed to analyze people's feelings and views on a specific subject. A plethora of studies has been conducted in various languages, with their prime focus being on monolingual languages. However, code-mixed languages are an exception to this rule. Very few code-mixed language pairings have been attempted in the past. In addition to sentiment analysis, the Forum for Information Retrieval Evaluation (FIRE) has carried out many code-switching tasks. The tasks include Code-Mixed Question Answering, sentiment analysis for code-mixed Indian languages (Patra, Das, and Das 2018; ICON 2017^b), POS tagging for code-mixed Indian SM (ICON 2016^c), and code-mixed entity extraction. Stance Detection (Utsav *et al.* 2020). Hate speech (Saroj, Chanda, and Pal 2020) and offensive content identification in code-mixed data (Chanda *et al.* 2021), Conversational hate speech identification (Chanda, Sheth, and Pal 2022) and information retrieval (Chanda and Pal 2023) from code-mixed data also explored. At the FIRE conference, a shared task for sentiment analysis of SM data was conducted in two language pairs. Around 12,000 and 2,500 tweets were published for training in Hindi–English and Bengali–English, respectively, these tweets were included in the dataset that was utilized for the shared task. The shared task's top-performing system combined SVM classification with the word and character level n-gram data. A lexicon-based approach (Sharma, Srinivas and Balabantaray 2015) was used for the task of Sentiment Analysis on Code-Mixed Hindi–English text on the FIRE 2013. The user comments from the public Facebook profiles of two prominent celebrities made up the dataset. SemEval 2020 organized a Sentiment Analysis in a Code-Switched Data competition (Patwa *et al.* 2020), included tweets in Hindi–English and Spanish–English pairs. In the Dravidian Languages, there is a scarcity of data for experimentation on code-mixed data.

Only a handful of datasets combine Kannada and English for sentiment analysis. A Kannada–English (Appidi *et al.* 2020) code-mixed dataset was created for emotion prediction. A Tamil–English (Patra *et al.* 2015) code-mixed dataset was created as a part of a shared task on Sentiment Analysis of Indian Languages (SAIL). The data were extracted from Twitter. The movie reviews in the Bengali–English data were classified using SVM and a Naive Bayes model (Mandal and Das 2018). subword-LSTM was used to address the noisy nature of the text (Joshi *et al.* 2016). Not only in India, but code-mixing can be seen worldwide. GLUECoS (Khanuja *et al.* 2020)—an evaluation benchmark for code-mixed text—and LinCE (Aguilar, Kar, and Solorio 2020)—a centralized benchmark for ten corpora, four distinct code-switched language pairs, and four tasks have been conducted successfully. Shared tasks have previously been part of code-switching workshops organized in connection with significant NLP conferences. Computational Approaches to

^b<https://ltrc.iit.ac.in/icon2017/>

^c<https://ltrc.iit.ac.in/icon2016/>

Code Switching's first and second workshops shared a challenge on Language Identification^d for a large number of language pairings (Nepalese–English, Spanish–English, Mandarin–English, and Modern Standard Arabic–Arabic dialects). In the fourth workshop, there was another shared assignment which was Machine Translation^e for many language combinations. Another English–Spanish tweet dataset was created as a benchmark for sentiment analysis (Vilares, Alonso, and Gómez-Rodríguez 2016). The labels for the positive, negative, and neutral classes are annotated based on SentiStrength. Later, this same group expanded its research to examine code-switching in monolingual (Vilares, Alonso, and Gómez-Rodríguez 2015) and multilingual (Vilares, Alonso, and Gómez-Rodríguez 2017) contexts. To assess the tone of a tweet regarding a movie, a traditional word probabilities-based method was used (Padmaja *et al.* 2020). The tweet was mixed with Telugu and English language. So the author transliterates each Roman word into its matching Telugu script and calculates the likelihood of the word in each class. (Kusampudi, Sathineni, and Mamidi 2021) introduced an annotated dataset for Sentiment Analysis on Code-Mixed Telugu-English Text (CMTET). Along with the dataset, they used a novel unsupervised data normalization method with a multilayer perceptron (MLP) model. Lee and Wang (2015) offer a proposal of a method to annotate the data gathered with emotions for a specific corpus of Chinese–English. The text that a feeling is expressed in is considered when developing the schema. This can be written in either Chinese or English, both languages, or a text that combines both. Another Chinese dataset was created by (Wang *et al.* 2015). The data was collected from Weibo.com (Wang *et al.* 2016). since the last 2-3 years, Dravidian code-mixed language has been getting special attention. Two Dravidian CodeMix sentiment analysis was organized at FIRE 2020^f and FIRE 2021.^g Apart from standard word features TF-IDF, FastText (Chanda and Pal 2020), Transformer based model, Multilingual word embedding, transfer learning, and meta embedding (Chanda, Singh and Pal 2021) were used to identify the sentiment of Dravidian language pairs. mBERT^h and XLM-Roberta-basedⁱ model achieved state-of-the-art performance. In this paper, we also used mBERT model as a baseline and compared its performance with the RQs.

3. Background

Here we set necessary technical background for further discussion. We briefly discuss some of the theoretical concepts, which have been used in further sections and different metrics used in measuring performances.

3.1 Multilingual BERT or mBERT

Word embeddings play a crucial role in natural language processing (NLP) tasks by representing words as dense numerical vectors in a continuous vector space. These embeddings capture the semantic and syntactic relationships between words, enabling machine learning models to leverage this contextual information for various language-related tasks. Traditionally, NLP models relied on one-hot encoding, where each word in a vocabulary was represented as a sparse binary vector. However, one-hot encoding does not capture the inherent semantic relationships between words. Word embeddings, on the other hand, provide a dense representation where similar words are located closer together in the vector space, reflecting their semantic similarity.

BERT (acronym for Bidirectional Encoder Representations from Transformers), a transformer architecture, functions as an encoder-decoder network employing self-attention on

^d<http://emnlp2014.org/workshops/CodeSwitch/call.html>

^e<https://code-switching.github.io/2021>

^f<https://dravidian-codemix.github.io/2020/>

^g<https://dravidian-codemix.github.io/2021/index.html>

^h<https://huggingface.co/bert-base-multilingual-cased>

ⁱ<https://huggingface.co/xlm-roberta-base>

the encoder side and attention on the decoder side. The model is pretrained on large text corpora such as Wikipedia and produce state-of-the-art results with necessary fine-tuning on several downstream tasks. The contextual language representation model BERT has been used for the downstream task of code-mixed language also. Multilingual BERT or mBERT ([bert-base-multilingual-cased^j](https://github.com/google-research/bert/blob/master/multilingual.md)) is pretrained on cased text in the top 104 languages with the largest Wikipedia and has a total of 179 M parameters with 12 transformers blocks, 768 hidden layers and 12 attention head. This model takes a special [CLS] token as input first, followed by a sequence of words as input. It then passes the input to the next layer. [CLS] here stands for Classification. Each layer applies self-attention and passes the result through a feedforward network to the next encoder.

3.2 Language identification tool by googletrans

We use Googletrans,^k a free and unlimited python library that implements Google Translate API. It uses the Google Translate Ajax API to make calls to such methods as detect (for language detection) and translate (to a target language).

3.3 Code-mixing metrics

Code-mixed dataset may have diverse code-mixing patterns. There are a few measures to quantify the extent of code-mixing, available in literature: Integration index (I-Index) (Barnett *et al.* 2000), multilingual index (M-Index) (Guzman *et al.* 2017), code mixed index (CMI) (Gambäck and Das 2014) etc. Among them, CMI is the most popular for its simplicity. CMI provides a straightforward and intuitive interpretation of the complexity of code-mixed data. It also compares well across datasets. We thus use CMI.

As per (Gambäck and Das 2014),

$$CMI = \frac{\sum_{i=1}^N (|W_i|) - \max \{|W_i|\}}{n - u}$$

where $\sum_{i=1}^N (|W_i|)$ is the sum of the number of words in all N languages present in an utterance, but not considering named entities (NEs) and other non-language symbols. $\max \{|W_i|\}$ is the highest number of words present in any language. n is the total number of tokens, and u is the number of tokens given other tags [here, for EN-TA dataset, KA (Kannada), and MA (Malayalam), Hindi (HI), etc. and so on.

The range of CMI is generally expressed in percentage, between 0 and 100 and is thus also written as follows.

$$CMI = \begin{cases} 100 \times \left[1 - \frac{\max\{|W_i|\}}{n-u} \right] & : n > u \\ 0 & : n = u \end{cases}$$

$CMI=0$ represents an utterance containing only monolingual words or only language-independent tokens.

An example from the Tamil–English dataset is shown in Figure 1. We can see that this is a classic instance of a single-language utterance. Each word from the sentence belongs to Tamil only, so the CMI value is zero.

Another example from the Tamil–English dataset is presented in Figure 2.

The provided sentence is code-mixed, comprising five words from Tamil and one word from English. Thus, code-mixing index (CMI) = $\frac{100 \times (6-5)}{6-0} = 16.66$ for this sentence.

^j<https://github.com/google-research/bert/blob/master/multilingual.md>

^k<https://pypi.org/project/googletrans/>



Figure 1. A monolingual example from Tamil-English dataset.

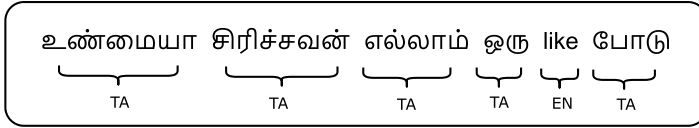


Figure 2. A code-mixed example from Tamil-English dataset.

3.4 Evaluation metrics

We evaluate and compare every model's performance in terms of precision, recall, F_1 -score, weighted average F_1 -score and accuracy for highly imbalanced label distribution, the definitions of which are listed below.

- **Precision:** It is the ratio of true-positives (TP) to the sum of TP and false-positives (FP).

$$\text{Precision}(P) = \frac{TP}{TP+FP}$$
- **Recall:** It is the ratio of true-positives (TP) to the sum of TP and false-negatives (FN).

$$\text{Recall}(R) = \frac{TP}{TP+FN}$$
- **F_1 -score:** It is the balanced harmonic mean of precision and recall and used to have a composite idea of precision and recall. $F_1\text{-score} = \frac{2 \cdot R \cdot P}{R+P}$
- **Macro avg F_1 :** It is the simple average of class-wise F_1 -scores. n is the number of class.

$$\text{Macro avg } F_1 = \frac{\sum_{i=1}^n F_1\text{-score}_i}{n}$$
- **Weighted avg F_1 :** It is the weighted version of the average F_1 -scores where each class is weighted by the number of samples from that class (also called support of that class).

$$\text{Weighted avg } F_1 = \frac{\sum_{i=1}^n \text{support}_i \cdot F_1\text{-score}_i}{\sum_{i=1}^n \text{support}_i}$$
- **Accuracy:** It is the ratio of number of correct predictions to the total number of test instances i.e., $\text{Accuracy} = \frac{\# \text{ correct predictions}}{\# \text{ test-instances}} = \frac{TP+TN}{TP+TN+FP+FN}$

4. Datasets

The Dravidian-CodeMix shared task¹ organizers provide a dataset comprising a training, a development and a test set. The organizers of the study disseminated the training and development datasets in the format of tab-separated values (.tsv) files. These files are structured with two distinct columns: “text” and “label.” Each individual row within these files represents a single data sample containing a youtube comment. Test dataset has an additionally “id” field for each data-item. Entire dataset maintained this format consistently across all three language pairs: Tamil-English (TA-EN) (Chakravarthi *et al.* 2020), Kannada-English (KA-EN) (Hande, Priyadharshini, and Chakravarthi 2020), and Malayalam-English (MA-EN) (Chakravarthi *et al.* 2020). The statistics of training, development and test data corpus collection and their class distribution are shown in

¹<https://dravidian-codemix.github.io/2021/index.html>

Table 1. Data distribution for sentiment detection of code-mixed text in Dravidian languages

TAMIL-ENGLISH				
Class	Training	Development	Test	Total
Positive	20070	2257	2546	24873
Negative	4271	480	477	5228
not-Tamil	1667	176	244	2087
Mixed_feelings	4020	438	470	4928
unknown_state	5628	611	665	6904
Total	35656	3962	4402	44020
KANNADA-ENGLISH				
Class	Training	Development	Test	Total
Positive	2823	321	374	3518
Negative	1188	139	157	1484
not-Kannada	916	110	110	1136
Mixed_feelings	574	52	65	691
unknown_state	711	69	62	842
Total	6212	691	768	7671
MALAYALAM-ENGLISH				
Class	Training	Development	Test	Total
Positive	6421	706	780	7907
Negative	2105	237	258	2600
not-Malayalam	1157	141	147	1445
Mixed_feelings	926	102	134	1162
unknown_state	5279	580	643	6502
Total	15880	1766	1962	19616

Table 1. The details of the dataset and benchmark results are given in overview (Priyadharshini *et al.* 2021) and findings (Chakravarthi *et al.* 2021) of the Sentiment Analysis of Dravidian Languages.

Even though entire the dataset is segregated as language pairs, there are examples of impurity: i.e., a TA-EN data may contain comments in Hindi (HI) or other languages. The dataset also suffers from other general problems of SM data, particularly code-mixed data like the sentences are short with a lack of well-defined grammatical structures and many spelling mistakes. The example texts from three language pairs are shown in Table 2.

Each dataset is labeled into five classes. The class labels are as follows.

- **Positive:** A comment comprises a textual cue, either explicit or implicit, that suggests the speaker is in a good frame of mind.

Table 2. Example of code-mixed text in Dravidian languages from three language pairs for all classes

Text	Label
TAMIL-ENGLISH	
வன்னியர் சாதி சார்பாக படம் வெற்றி பெற வாழ்த்துக்கள்....	Positive
Il FAUT que WTC voit ca	not-Tamil
வெந்து வேகதா அரை வேகடு ஹீரோது	Negative
தல போல வராது.... Waiting viswasam trailer	Mixed_feelings
Rajini fans comment. Neutral fans like.	unknown_state
KANNADA-ENGLISH	
Tiktokers present situation. . . ನೋಡುವವರು ಯಾರು ನಮ್ಮ ವೀಡಿಯೋನೂ	Negative
I do like Guru Deshpande Movies. Im gonna rush	not-Kannada
ಮನೆಯಿಂದಲೇ ಸಂಪಾದಿಸಬೇಕೇ? With income proof. WhatsApp 6362*****	unknown_state
Super ಸಾಂಗ್ ವೆರಿ ಸ್ಪೆಸ್....	Positive
@Wild Rex ಕಟ್ಟಬೇಕು bronಖಂಡಿತಾ ಕಟ್ಟುತ್ತೆ bro	Mixed_feelings
MALAYALAM-ENGLISH	
ഹഹദിന്റെ ഭാര്യായിറ്റ് അഭിനയിച്ച ചേച്ചി ഔട്സ്റ്റാൻഡിങ് ആക്ടിങ്	Positive
ആമയെ വെള്ളത്തിൽ മുക്കി കൊല്ലുന്ന ഇനമാ.. ..	unknown_state
yenthu chali annu bro ethu	Negative
nyc but best actor poley aayaaal seriyaavilla	Mixed_feelings
The face of indiaaikkkkkkkaaaaaa luv uuuuuu	not-Malayalam

- **Negative:** Comments contain an explicit or implicit clue in the text suggesting that the speaker is in a negative state.
- **Mixed_feelings:** Comments contain an explicit or implicit clue in both positive and negative feelings.
- **unknown_state:** The comment does not contain an explicit or implicit indicator of the speaker’s emotional state.
- **Not in intended language:** For Tamil, if the sentence does not contain Tamil written in Tamil script or Roman script, then it is not-Tamil.

5. Methodology and experiment setup

This section provides a detailed description of the methodology and experimental setup employed in this study for the different RQs presented in Section 1.

5.1 Data preprocessing

The dataset provided was not very clean and required preprocessing. The preprocessing pipeline contains several steps to ensure the cleanliness and standardization of the text data:

- In order to mitigate the effect of repeated characters, a restriction is imposed wherein only two consecutive repetitions are allowed. For instance, the word “goooooooood” is simplified to “good”.

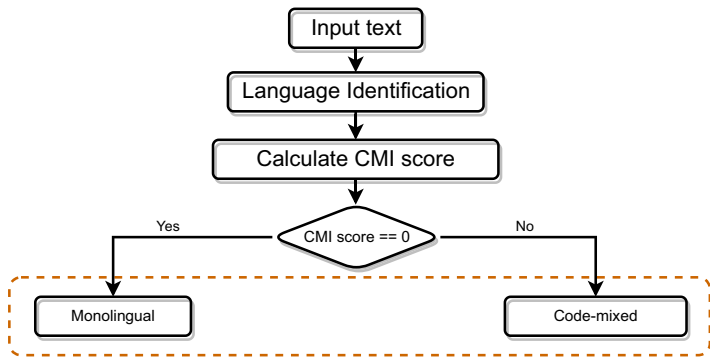


Figure 3. Model Architecture for identifying monolingual and code-mixed data.

- Hashtags and URLs are eliminated from the text.
- Exclamations and other punctuation marks are removed.
- Non-ASCII characters, symbols, numbers, and special characters are all excluded.
- Emojis are replaced with their corresponding semantic text.
- Any trailing or excessive white spaces are stripped off, ensuring a uniform text format.

The cleaned data is then subject to different steps as the RQs demanded. We detail the methodology adopted for each such RQ below.

5.2 Methodology for RQ-1

In order to check if the dataset is code-mixed or not, we apply word-level language identification to determine which languages are involved in a utterance. We deploy a language identification (LID) module (implemented using Googletrans), which provides word-level language identification for each word. We see the number of constituent languages used in a sentence and the frequency with which a user changes languages in a given sentence and CMI is calculated for each sentence. All the sentences in the dataset are then divided into two categories. The sentences having a CMI value of zero are monolingual. The rest are classified as code-mixed sentences (see Figure 3). Subsequently, we train the model on training and validation data separately on monolingual and code-mixed data.

Our initial objective is to first investigate the impact of training using different components of data from the given datasets: using

1. all the original data (code-mixed and monolingual together)
2. using only code-mixed data
3. employing only monolingual data

Separation of (2) and (3) is done using CMI as discussed above. Subsequently, we conduct experiments using test datasets on each of the cases above.

The given task here is seen as single-stage classification with five classes.

The sentences are converted to vectors using word embedding according to (mBERT) pre-trained model (Pires, Schlinger, and Garrette 2019) to get a vector as an embedding for the sentence to be used for classification. On top of that, we also use a multilayer perceptron (MLP) where we freeze the parameters of the pretrained model and generate the prediction.

For the implementation, HuggingFace's transformers library^m is utilized. HuggingFace is a Python package that offers pretrained and adaptable transformer models to be used for various NLP tasks. The implementation environment is the PyTorch library, which supports GPU computation. The mBERT models are run using Google Colab. We train our classifier 10 epochs with a batch size of 32. The AdamW optimizer is used, and the dropout value is set at 0.1. The learning rate is $2e-5$. For tokenization, we utilize the hugging face transformers' pretrained BERT tokenizer. We utilize the HuggingFace library's BertForSequenceClassification module for tinkering and sequence classification.

5.3 Methodology for RQ-2

In RQ-1, we considered sentiment analysis task as a multi-class classification problem. However, for the given dataset, one of the classes is "Not in intended language", that, according to the original description of the dataset, is any statement which does not contain a word from the specified language. This class needs language identification while the other four are based on sentiment identification. In RQ-2, we hypothesize that this language identification should be used directly in classification. While a classifier can be used for finding the sentiments, another can be used for finding the language/ not language part. Consequently, we approach it as a multi-class classification problem like in RQ-1. But additionally we use another language identification (LID) module to identify not in language to augment the classification task.

For the multi-class classification, we use the mBERT model which we already discuss in the previous section. We use the combined (monolingual and code-mixed) dataset for training, development and testing for this experiment.

For the language identification (LID), like before, we use Googletrans API to decide whether a sentence is in <language> or 'not-<language>'. The mBERT predictions as sentimental tags (positive, negative, mixed_feelings or unknown_state) are kept only if the output is <language>, otherwise, are marked as 'not-<language>' and added to the class of 'not-<language>' by mBERT as well (see Figure 4).

5.4 Methodology for RQ-3

In this experiment, we develop our system in a hierarchical fashion. First, we use the LID module to determine <language> and one "not-<language>". All the sentences labeled as <language> only by the LID-module are fed to the mBERT module for sentiment classification. The "not-<language>" are solely decided by the LID module, unlike in RQ-2 (see Figure 5).

For this experiment, we utilize the combined dataset (consisting of monolingual and code-mixed samples) for training, development, and testing purposes.

6. Results and discussion

This section reports experimental results conducted to address the RQs on the three language pairs followed by discussion on the findings. Furthermore, in Section 6.4, an in-depth exploration into error analysis is also undertaken.

6.1 Results for RQ-1

The primary objective of RQ-1 is to check whether the data are code-mixed or not. It is clear from Table 3 that the dataset contains a large percentage of monolingual data. In the Tamil-English

^mhttps://huggingface.co/transformers/pretrained_models.html

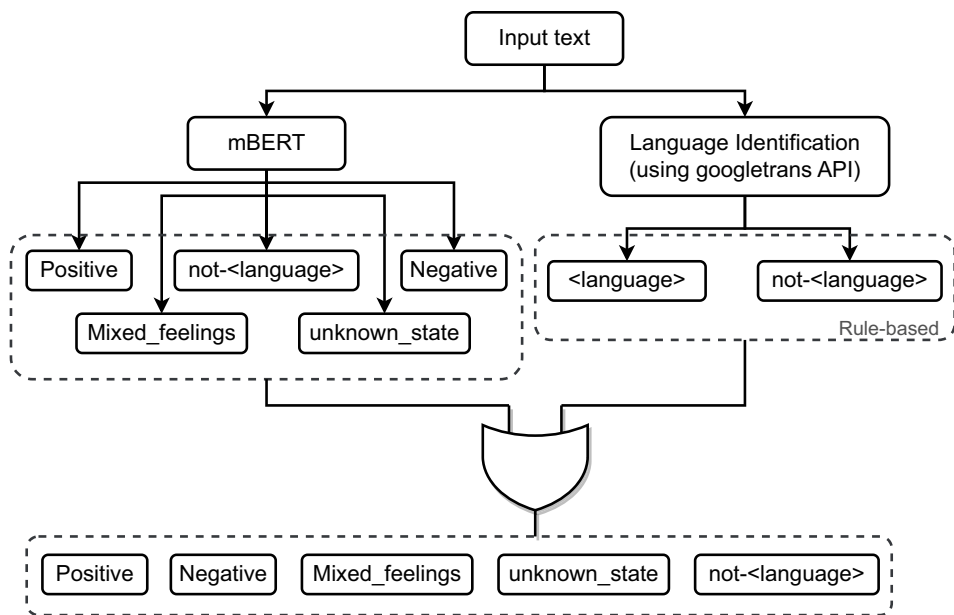


Figure 4. Model architecture for multi-class classification with ruled-based language tag.

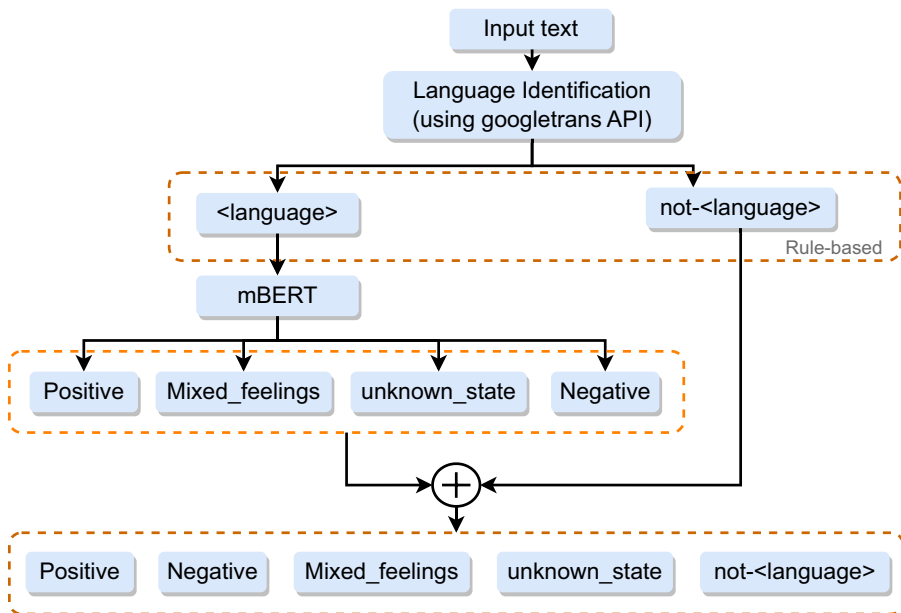


Figure 5. Model architecture for hierarchical approach with mBERT.

dataset, more than 40% of the data are monolingual in the three distributions: training, development, and test. In the Malayalam–English dataset, the percentage is less in all three language pairs. 35.8%, 37.5%, and 34.5% of the data are monolingual in training, development, and test, respectively. However, share of monlingual data is the highest in Kannada–English: 57.4%, 57.4%, and 56.6% in training, development, and test data, respectively.

Table 3. The statistics of monolingual and code-mixed data involved in training, development, and test datasets of all three language pairs

Tamil-English					
Training		Development		Test	
Monolingual	Code-mixed	Monolingual	Code-mixed	Monolingual	Code-mixed
14385	21271	1588	2374	1783	2619
Kannada-English					
Training		Development		Test	
Monolingual	Code-mixed	Monolingual	Code-mixed	Monolingual	Code-mixed
3568	2644	397	294	435	333
Malayalam-English					
Training		Development		Test	
Monolingual	Code-mixed	Monolingual	Code-mixed	Monolingual	Code-mixed
5702	10186	664	1102	676	1286

Table 4. Level of code-mixing (CMI values) involved in training, development, and test datasets of all three language pairs

	Tamil-English		Kannada-English		Malayalam-English	
	CMI-All	CMI-Mixed	CMI-All	CMI-Mixed	CMI-All	CMI-Mixed
Train	18.39	30.83	13.67	32.13	19.16	29.89
Dev	18.63	31.09	14.45	33.97	18.82	30.16
Test	18.38	30.9	13.5	31.13	20.09	30.65

Table 4 depicts the code-mixing level involved in each distribution of three language pairs. We report the average CMI values in two ways: first, considering the original dataset including code-mixed and monolingual (represented as CMI-All) and then considering only the pure code-mixed ones (discarding the monolingual ones, represented as CMI-Mixed). Since the given datasets contain a substantial share of monolingual sentences having CMI values zero, the scores increase from CMI-All to CMI-Mixed as these values are discarded from calculation in the second.

We see the impact of different types of training data: all original data with monolingual (mono) and code-mixed (CM) combined; only code-mixed data; and only monolingual data for 3 language-pairs on testing with all data (monolingual + code-mixed). The performance of our proposed models is shown in terms of precision, recall, macro averaged F_1 -score, and weighted average F_1 -score in Tables 5, 6, and 7, respectively.

The findings reveal a performance decline for all three language pairs when we exclusively employ either code-mixed or monolingual data only for training. For instance, in the case of Malayalam-English, the weighted average F_1 score was 0.69. However, when we train the model solely on code-mixed data and conduct test on the combine dataset, the weighted average F_1 score decrease to 0.66, indicating a performance drop of approximately 4% (Table 7). Furthermore, when we employ monolingual data for training and test the model on the combine dataset, the weighted average F_1 score decrease by 14%, from 0.69 to 0.59.

Table 5. Precision, recall, F_1 -scores, and support for all experiments on Tamil-English test data

	Train on Mono and CM data			Train on CM data			Train on Mono data			support
	Precision	Recall	F_1 -score	Precision	Recall	F_1 -score	Precision	Recall	F_1 -score	
Mixed_feelings	0.26	0.26	0.26	0.22	0.23	0.22	0.22	0.21	0.21	470
Negative	0.40	0.34	0.36	0.38	0.31	0.34	0.36	0.25	0.29	477
Positive	0.75	0.79	0.77	0.75	0.75	0.75	0.72	0.81	0.76	2546
not-Tamil	0.65	0.53	0.58	0.61	0.46	0.52	0.59	0.45	0.51	244
unknown_state	0.40	0.38	0.39	0.37	0.43	0.40	0.38	0.33	0.35	665
macro avg	0.49	0.46	0.47	0.47	0.44	0.45	0.45	0.41	0.43	4402
weighted avg	0.60	0.61	0.60	0.59	0.58	0.58	0.57	0.59	0.58	4402
Accuracy	0.61			0.58			0.59			

Table 6. Precision, recall, F_1 -scores, and support for all experiments on Kannada-English test data

	Train on Mono and CM data			Train on CM data			Train on Mono data			support
	Precision	Recall	F_1 -score	Precision	Recall	F_1 -score	Precision	Recall	F_1 -score	
Mixed_feelings	0.23	0.28	0.25	0.12	0.14	0.13	0.11	0.12	0.12	65
Negative	0.66	0.54	0.59	0.59	0.50	0.54	0.65	0.52	0.58	157
Positive	0.70	0.74	0.72	0.70	0.72	0.71	0.70	0.76	0.73	374
not-Kannada	0.59	0.59	0.59	0.53	0.55	0.54	0.62	0.59	0.61	110
unknown_state	0.31	0.29	0.30	0.24	0.24	0.24	0.33	0.29	0.31	62
macro avg	0.50	0.49	0.49	0.44	0.43	0.43	0.48	0.46	0.47	768
weighted avg	0.61	0.60	0.60	0.57	0.56	0.56	0.60	0.60	0.60	768
Accuracy	0.60			0.56			0.60			

These outcomes suggest that utilizing solely code-mixed or monolingual data for training adversely affects the model’s performance on the combined test dataset, which consists of both code-mixed and monolingual samples.

We also look at the performance when the system is trained on exclusively pure code-mixed data, and separate tests are conducted on monolingual, code-mixed, and combined datasets. Table 8 presents the weighted average F_1 scores for three language pairs. (We chose weighted average F_1 to summarize the performances without going to the details.) Notably, the performance of Kannada-English and Malyalam-English language pairs exhibit a decline when utilizing both the combined and monolingual test data. However, the overall performance for TN-EN pair is not hampered.

It is essential to consider both the weighted average F_1 score and the support value as crucial metrics. A higher weighted average F_1 score signifies superior overall performance, while a larger support value indicates higher number of data points. This implies that a larger dataset was used, which could be the result of collecting data from a larger population, a longer time span, or a more

Table 7. Precision, recall, F_1 -scores, and support for all experiments on Malayalam–English test data

	Train on Mono and CM data			Train on CM data			Train on Mono data			support
	Precision	Recall	F_1 -score	Precision	Recall	F_1 -score	Precision	Recall	F_1 -score	
Mixed_feelings	0.40	0.16	0.23	0.34	0.31	0.32	0.22	0.14	0.17	134
Negative	0.55	0.53	0.54	0.56	0.44	0.49	0.40	0.37	0.38	258
Positive	0.74	0.81	0.77	0.72	0.78	0.75	0.64	0.75	0.69	780
not-Malayalam	0.83	0.71	0.77	0.78	0.63	0.70	0.61	0.72	0.66	147
unknown_state	0.71	0.74	0.72	0.67	0.71	0.69	0.67	0.58	0.62	643
macro avg	0.64	0.59	0.61	0.61	0.57	0.59	0.51	0.51	0.51	1962
weighted avg	0.69	0.70	0.69	0.66	0.67	0.66	0.59	0.60	0.59	1962
Accuracy	0.70			0.67			0.60			

Table 8. Weighted average F_1 -scores and support for three language pairs where model is trained only on CM data but tested on all, Monolingual and CM data

Language pair	Test on all data		Test on monolingual data		Test on code-mixed data	
	Weighted avg. F_1 -score	support	Weighted avg. F_1 -score	support	Weighted avg. F_1 -score	support
Tamil–English	0.58	4402	0.60	1783	0.58	2619
Kannada–English	0.56	768	0.53	435	0.60	333
Malayalam–English	0.66	1962	0.61	676	0.69	1286

extensive sampling process. The utilization of a larger dataset provides a broader range of information, potentially capturing more diverse scenarios, variations. The observations depicted in Table 8 indicate that, relative to monolingual data, the models perform marginally better when subjected to tests using code-mixed data. Regarding the Tamil–English language pairs, the performance of the models exhibits a comparable trend on both code-mixed and combined datasets, with monolingual data showing promising results. Nevertheless, when considering the support value, the model achieves optimal performance on combined datasets, followed by code-mixed data and then monolingual data. In the case of the Kannada–English and Malayalam–English language pairs, conducting tests on code-mixed data yields favorable outcomes in comparison to monolingual and combined datasets. In the specific scenario of Malayalam–English language pairs, training and testing the model on code-mixed data lead to a weighted average F_1 score of 0.69. However, training on code-mixed data and testing on the complete dataset result in a weighted average F_1 score of 0.66, indicating a decline in performance of approximately 4%. Remarkably, utilizing monolingual data for testing purposes further decreases the weighted average F_1 score by 11.5%, from 0.69 to 0.61.

6.2 Results for RQ-2

Here we use the mBERT model to forecast the sentiment polarity for the Tamil–English, Malayalam–English, and Kannada–English languages. However, we also use a LID module followed by a rule-based approach to divide sentences into two categories: language and

Table 9. Comparison of Precision, recall, F_1 -scores, and support with our proposed model for RQ-2 on Tamil–English test data

	mBERT			mBERT + Ruled-based systems			
	Precision	Recall	F_1 -score	Precision	Recall	F_1 -score	Support
Mixed_feelings	0.26	0.26	0.26	0.31	0.19	0.23	470
Negative	0.40	0.34	0.36	0.44	0.39	0.41	477
Positive	0.75	0.79	0.77	0.76	0.83	0.80	2546
not-Tamil	0.65	0.53	0.58	0.79	0.98	0.87	244
unknown_state	0.40	0.38	0.39	0.45	0.43	0.44	665
macro avg	0.49	0.46	0.47	0.55	0.56	0.55	4402
weighted avg	0.60	0.61	0.60	0.63	0.66	0.64	4402
Accuracy	0.61			0.66			

Table 10. Comparison of precision, recall, F_1 -scores, and support with our proposed model for RQ-2 on Kannada–English test data

	mBERT			mBERT + Ruled-based systems			
	Precision	Recall	F_1 -score	Precision	Recall	F_1 -score	Support
Mixed_feelings	0.23	0.28	0.25	0.21	0.28	0.24	65
Negative	0.66	0.54	0.59	0.71	0.54	0.61	157
Positive	0.70	0.74	0.72	0.77	0.76	0.76	374
not-Kannada	0.59	0.59	0.59	0.71	1.00	0.83	110
unknown_state	0.31	0.29	0.30	0.39	0.24	0.30	62
macro avg	0.50	0.49	0.49	0.56	0.56	0.55	768
weighted avg	0.61	0.60	0.60	0.67	0.67	0.66	768
Accuracy	0.60			0.67			

not-<language>. Finally, we combine these predictions with the mBERT prediction and then provide prediction.

Tables 9, 10, and 11 display the performance of the test outcomes for the mBERT model and mBERT + Ruled-based systems. For all three language pairs, we can observe a performance increase in terms of weighted average F_1 scores. Our model witnessed an increase of 6.6%, 10%, and 4.3% on Tamil–English, Kannada–English, and Malayalam–English, respectively. In this study, we wanted particularly to improve the not-language tag performance with the addition of LID module which we can clearly observe for all language pairs. For Tamil–English language, F_1 score of not-Tamil class improve from 0.58 to 0.87 almost 50%. For Kannada–English language, F_1 score of not-Kannada class improve from 0.59 to 0.83 almost 40%. For Malayalam–English language, F_1 score of not-Malayalam class improve from 0.77 to 0.89 almost 15%.

Not only the not-<language> class, we also see slight performance improvement in other classes as well. The reason behind is that mBERT sometimes wrongly classifies not-<language> as other classes which create a large set of false positives. When these are removed as detected by the (LID + rule-based) system, finally the precision value of all classes across the language pairs

Table 11. Precision, recall, F_1 -scores, and support with our proposed model for RQ-2 on Malayalam–English test data

	mBERT			mBERT + Ruled-based systems			support
	Precision	Recall	F_1 -score	Precision	Recall	F_1 -score	
Mixed_feelings	0.40	0.16	0.23	0.43	0.25	0.32	134
Negative	0.55	0.53	0.54	0.63	0.56	0.59	258
Positive	0.74	0.81	0.77	0.77	0.82	0.79	780
not-malayalam	0.83	0.71	0.77	0.81	0.97	0.89	147
unknown_state	0.71	0.74	0.72	0.72	0.74	0.73	643
macro avg	0.64	0.59	0.61	0.67	0.67	0.66	1962
weighted avg	0.69	0.70	0.69	0.72	0.73	0.72	1962
Accuracy	0.70			0.73			

Table 12. Comparison of precision, recall, F_1 -scores, and support with our proposed model for RQ-3 on Tamil–English test data

	mBERT				mBERT + Hierarchical Model			
	Precision	Recall	F_1 -score	Support	Precision	Recall	F_1 -score	Support
Mixed_feelings	0.26	0.26	0.26	470	0.36	0.20	0.25	447
Negative	0.40	0.34	0.36	477	0.39	0.36	0.37	447
Positive	0.75	0.79	0.77	2564	0.74	0.85	0.80	2305
unknown_state	0.40	0.38	0.39	665	0.41	0.35	0.38	590
macro avg	0.45	0.44	0.44	4176	0.48	0.45	0.47	3789
weighted avg	0.59	0.61	0.60	4176	0.60	0.64	0.62	3789

improves. Overall, mBERT + Rule-based model outperforms the mBERT model for all three language pairs.

6.3 Results for RQ-3

Here we propose a hierarchical model. In the first level of hierarchical model, we use a LID module followed by a rule-based approach to divide sentences into two categories: <language> and not-<language>. In the second level, all the sentences labeled as <language> are sent to mBERT module for sentiment classification among four classes: positive, negative, unknown_state, and mixed_feelings.

Tables 12, 13, and 14 display the result of the test data for the mBERT model and mBERT + Hierarchical model. For all the three language pairs, we observe a performance gain in terms of weighted average F_1 scores. Our model witness an increase of 3.33%, 11.6% and 6% on Tamil–English, Kannada–English, and Malayalam–English, respectively.

We observe a gross anomaly after the first-level of language identifications: <language> and not-<language>. A substantial number of sentences classified into not-<language> classes are wrongly shown belonging to other classes in the gold standard data (for details, see below). Our method

Table 13. Comparison of precision, recall, F_1 -scores, and support with our proposed model for RQ-3 on Kannada–English test data

	mBERT				mBERT + Hierarchical Model			
	Precision	Recall	F_1 -score	Support	Precision	Recall	F_1 -score	Support
Mixed_feelings	0.23	0.28	0.25	65	0.39	0.12	0.19	56
Negative	0.66	0.54	0.59	157	0.70	0.54	0.61	154
Positive	0.70	0.74	0.72	374	0.77	0.76	0.76	312
unknown_state	0.31	0.29	0.30	62	0.37	0.38	0.37	37
macro avg	0.46	0.44	0.46	658	0.54	0.48	0.52	559
weighted avg	0.60	0.61	0.60	658	0.66	0.68	0.67	559

Table 14. Comparison of precision, recall, F_1 -scores, and support with our proposed model for RQ-3 on Malayalam–English test data

	mBERT				mBERT + Hierarchical Model			
	Precision	Recall	F_1 -score	support	Precision	Recall	F_1 -score	Support
Mixed_feelings	0.40	0.16	0.23	134	0.38	0.23	0.29	129
Negative	0.55	0.53	0.54	258	0.56	0.50	0.53	252
Positive	0.74	0.81	0.77	780	0.74	0.82	0.78	745
unknown_state	0.71	0.74	0.72	643	0.71	0.71	0.71	613
macro avg	0.60	0.56	0.56	1815	0.60	0.57	0.58	1739
weighted avg	0.67	0.66	0.66	1815	0.69	0.70	0.70	1739

identifies 430 sentences as not-Tamil for the Tamil-English test dataset, whereas the gold data only indicates 244 instances.

This causes a decrease in the support count across all classes. In Table 12, we specifically observe that 23 sentences are classified as Mixed_feelings, which should be classified as not-Tamil sentences. The same thing happens for other classes also, like 30 from the negative class, 259 from the positive class, and 75 from the unknown_state class.

Despite such mis-classifications, our model’s weighted average F_1 score improves compared to the mBERT model. This improvement indicates that our model’s performance is becoming more accurate and reliable when apply to the test data.

Similar patterns are observed in Table 13 for the Kannada–English dataset, where our propose model consistently exhibited superior performance compare to mBERT. For the Malayalam–English dataset, as depicted in Table 14, the performance trends are similar.

On closer analysis, we discover that our LID-module divides the sentences into <language>and not-<language>correctly for a majority of the cases. The deviation from the actual tags is primarily due to the error in the annotation. However, there are some errors made by the LID module as well discussed below.

6.4 Error analysis

We categorize the errors into two types: errors in annotation of gold standard and errors made by our model.

Table 15. Errors in gold standard

No.	Sample text from dataset	Gold	Predicted
1	Govt should take action	neutral	not-Tamil
2	Police pls take action	neutral	not-Tamil
3	Government take this type of peoples civiliar actions	neutral	not-Tamil
4	JJ mam we miss u	Positive	not-Tamil
5	Kangana Ranaut ke fans like here	unknown_state	not-Tamil
6	Funny to hear tamil language for urdu/punjabi speaking people. at puta nagada duta mala keda under duka etc. lol	not-Tamil	Tamil
7	Super song bro	Positive	not-Kannada
8	@Ranganath S yes	unknown state	not-Kannada
9	and rocking star fans also	Positive	not-Kannada
10	YouTube logic 66k likes + 6.3k dislikes = 16k views..	Negative	not-Malayalam

Table 16. Errors made by the LID model

No.	Sample text from dataset	Gold	Predicted
1	Lv u rrr shetty super	not-Kannada	Kannada
2	Pora rai koodi tulla	not-Kannada	Kannada
3	One an only megastar mammookka	not-Malayalam	Malayalam
4	aa musicinu vendi wait cheyyuva	not-Malayalam	Malayalam
5	300 crores next. Ok alle	unknown_state	not-Malayalam
6	MEGA STAR FANS undo like aadiki	unknown_state	not-Malayalam
7	4M viewsmammookka we are waiting	not-Malayalam	Malayalam

Table 15 presents few instances where our model made correct predictions, but annotations in the gold standard are wrong. Column Gold shows the annotations of gold standard, while column Predicted shows results of our model. As we can see that the labels predicted by our model align more accurately with the reality rather than in the gold standard. From examples 1 to 5, It is clear that all of the sentence does not contain any Tamil words, so this should be comes under not-Tamil class. For example 6, despite the presence of many Tamil words according to the LID module, the gold standard label it as not-Tamil, contradicting the reality.

LID module plays a significant role in cases where our model failed to predict the correct classes. Table 16 presents instances where our model made incorrect predictions. For instance, in Example 1, the LID module labels “shetty” as a Kannada word possibly because the word is an out-of-vocabulary (OOV) word, leading our model to predict it as Kannada. However, this is a name of a person, and other words are English. Example 2 is actually a Tamil expression, but the LID module identifies it as Kannada. In Example 3, “mammookka” was labeled as a Malayalam word by the LID module, while it is a name. In Example 5, the word “alle,” is a Malayalam word, but labeled as Italian by the LID module.

In general, we observe that LID module struggled to accurately identify the proper language tag for NEs. Consequently, the performance of the word-level language identification module was inadequate, as it failed to detect the appropriate tags. In the Tamil–English dataset alone, we encountered a total of 28 language tags, prompting us to develop a rule-based model where we designated these tags as unknown tags.

These findings emphasize the limitations of the LID module and the challenges it poses in accurately determining the language for certain words and sentences.

6.5 Discussion

The findings from our research provide valuable insights into sentiment analysis in code-mixed languages and offer practical implications for handling such datasets effectively. The three RQs are interconnected and contribute collectively to a comprehensive understanding of SA in code-mixed datasets. RQ-1 underscores the significance of dataset composition, with the combination of code-mixed and monolingual data influencing model performance. In RQ-1, we apply a state-of-the-art model and report the best scores that are used later as the baselines to compare with the other RQs. RQ-2 highlights an important role of LID in code-mixed data processing, to further enhance the ability of the sentiment analysis model to handle multiple languages effectively. Tables 9, 10, and 11 display the results of the test data for the mBERT model (RQ-1) and mBERT + Rule-based model (RQ-2). On comparison between RQ-1 and RQ-2, we observe a noticeable performance gain for RQ-2 in terms of weighted average F_1 scores. RQ-3 explores a hierarchical approach that incorporates both LID and SA, showcasing improved performance and the complementary nature of these tasks in code-mixed datasets. Tables 12, 13, and 14 display the result of the test data for the mBERT model (RQ-1) and mBERT + Hierarchical model (RQ-3). For all three language pairs, we observe a performance gain for RQ-3 in terms of weighted average F_1 scores. Between RQ-2 and RQ-3, the main difference is whether LID should be used in parallel to classifier (RQ-2) or should be used first followed by the classifier (RQ-3) to augment the classification task. For the given datasets, experiments for RQ-2 provide comparable performances to RQ-3. However, real comparison between RQ-2 and RQ-3 is only possible if the datasets are free of annotation errors and performance of LID module is reasonably high. At the moment, with the amount of annotation errors, it is premature to comment on the comparative performances of experiments on RQ-2 and RQ-3.

7. Conclusion

The exponential growth of data generated by SM users in the era of Web 2.0 has necessitated advanced techniques for sentiment analysis. While numerous studies have focused on sentiment analysis in monolingual datasets, there is a scarcity of research in code-mixed datasets. In this paper, we present a methodology for identifying sentiment polarities in code-mixed languages using a dataset of Tamil–English, Kannada–English, and Malayalam–English YouTube SM comments. Our approach goes beyond traditional models by incorporating novel characteristics, such as the language tag module, which prove to yield improved results in the majority of cases.

Specific to our dataset, we draw the following conclusions for the RQs in the introduction.

RQ-1: When to call a dataset code-mixed? If a given dataset contains a mixture of pure monolingual, transliterated monolingual, single-script multilingual, and mixed script multilingual text, what is the best strategy for conducting sentiment analysis?

The dataset comprises a combination of code-mixed and monolingual sentences. We find that if a dataset contains a mix of code-mixed and monolingual data, it is preferable to train the model using the same type of data. Training the model solely on code-mixed data resulted in a performance drop. However, when the text data consist of exclusively of code-mixed content,

training the model using code-mixed data leads to better results. Therefore, the combination of code-mixed and monolingual data is a crucial factor to consider.

RQ-2: How important is the language identification (LID) task for code-mixed data processing? Can a separate language identification model with a pretrained model improve the overall system performance? We seek to find the answer particularly, with respect to sentiment analysis task.

Language Identification is a very important step. Across all three language pairs, our model demonstrated improved performance in terms of weighted average F_1 scores. Specifically, we observe performance increases of 6.6%, 10%, and 4.3% for Tamil–English, Kannada–English, and Malayalam–English, respectively, after incorporating a separate LID module that augments a simple, single-level training-based approach.

RQ-3: Is it possible to see SA as a multilevel problem rather than a multi-class problem? If yes, can a multilevel hierarchical model enhance the performance?

We explored a hierarchical model with LID and mBERT module. Across all three language pairs, our model showcased enhanced performance in terms of weighted average F_1 scores. Notably, we achieve performance increases of 3.33%, 11.6%, and 6% for Tamil–English, Kannada–English, and Malayalam–English, respectively.

Language identification and sentiment analysis are two independent tasks. But in case of code-mixed data, these two intertwined. However, to improve the performance of SA, LID is an important step—that should be taken separately. Our experiments could not show clearly whether LID should be done in parallel to augment the classification task (RQ-2) or it should be done first followed by SA on in-language data (RQ-3) because of annotation errors in the gold standard data. We hypothesize that the second should yield better performance if the performance of LID is reasonably good and the gold standard data is error free. However, this needs to be experimentally validated and it requires an error-free data. We are in the process of creating one such datasetⁿ and look forward to taking up the study in future.

Wrong predictions made by our model can be attributed to our reliance on “Googletrans,” a word-level language identification module, which is not entirely accurate and thus affects our evaluation scores. We acknowledge the need for further research and development of our own word-level identification model to improve the performance scores in the future. Furthermore, it is important to acknowledge that the system’s performance can be significantly influenced by the extent of code-mixing. The observed substantial disparity between CMI-Mixed and CMI-All values indicates the presence of numerous monolingual sentences, with varying degrees of code-mixing across different sentences. This aspect opens up a promising avenue for future research, wherein investigations could be conducted on the impact of distinct levels of code-mixing in sentences on downstream tasks. Such analyses would contribute valuable insights to the field.

References

- Aguilar G., Kar S. and Solorio T. (2020). LinCE: A centralized benchmark for linguistic code-switching evaluation. In *Proceedings of The 12th Language Resources and Evaluation Conference*, Marseille, France: European Language Resources Association, pp. 1796–1806. <https://www.aclweb.org/anthology/2020.lrec-1.222>.
- Appidi A. R., Srirangam V. K., Suhas D. and Shrivastava M. (2020). Creation of corpus and analysis in code-mixed Kannada-English Twitter data for emotion prediction. In *Proceedings of the 28th International Conference on Computational Linguistics, International Committee on Computational Linguistics*, Barcelona, Spain (Online), pp. 6703–6709.
- Bali K., Sharma J., Choudhury M. and Vyas Y. (2014). “I am borrowing ya mixing ?” An analysis of English-Hindi code mixing in Facebook. In *Proceedings of the First Workshop on Computational Approaches to Code Switching*, Doha, Qatar: Association for Computational Linguistics, pp. 116–126.
- Balouchzahi F. and Shashirekha H.L. (2021). LA-SACo: A study of learning approaches for sentiments analysis in Code-mixing texts. In *Proceedings of the First Workshop on Speech and Language Technologies for Dravidian Languages*, Kyiv. Association for Computational Linguistics, pp. 109–118.
- Barman U., Das A., Wagner J. and Foster J. (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*, Doha, Qatar:

ⁿ<https://malayalam-data.netlify.app/>

- Association for Computational Linguistics, pp. 13–23. <https://doi.org/10.3115/v1/W14-3902>. <https://www.aclweb.org/anthology/W14-3902>
- Barnett B., Codo E., Eppler E., Forcadell M., Gardner-Chloros P., van Hout R., Moyer M., Torras M., Turell M., Sebba M., Starren M. and Wensink S. (2000). The lides coding manual - A document for preparing and analyzing language interaction data. *International Journal of Bilingualism* 4(2), 131–270, version 1.1. Pagination: 140. 1999-07.
- Chakravarthi B. R., Jose N., uryawanshi S., Sherly E. and McCrae J. P. (2020). A sentiment analysis dataset for code-mixed Malayalam-English. 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)*, European Language Resources Association, Marseille, France, pp. 177–184.
- Chakravarthi B. R., Muralidaran V., Priyadharshini R. and McCrae J. P. (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)*, European Language Resources Association, Marseille, France, pp. 202–210.
- Chakravarthi B. R., Priyadharshini R., Muralidaran V., et al. (2022). DravidianCodeMix: sentiment analysis and offensive language identification dataset for Dravidian languages in code-mixed text. *Lang Resources & Evaluation* 56(3), 765–806. <https://doi.org/10.1007/s10579-022-09583-7>.
- Chakravarthi B. R., Priyadharshini R., Thavareesan S., Chinnappa D., Thenmozhi D., Sherly E., McCrae J. P., Hande A., Ponnusamy R., Banerjee S. and Vasantharajan C. (2021). Findings of the sentiment analysis of Dravidian languages in code-mixed text. In *Working Notes of FIRE. 2021 - Forum for Information Retrieval Evaluation*. CEUR.
- Chanda S. and Pal S. (2020). IRLab@IITBHU@Dravidian-codeMixFIRE2020: sentiment analysis for dravidian languages in code-mixed text. In *FIRE (Working Notes)*. Hyderabad, India: CEUR.
- Chanda S. and Pal S. (2023). The effect of stopword removal on information retrieval for code-mixed data obtained via social media. *SN Computer Science* 4(5), 494. <https://doi.org/10.1007/s42979-023-01942-7>.
- Chanda S., Sheth S. and Pal S. (2022). Coarse and fine-grained conversational hate speech and offensive content identification in code-mixed languages using fine-tuned multilingual embedding. In *Forum for Information Retrieval Evaluation (Working Notes) (FIRE)*. CEUR-WS.org.
- Chanda S., Singh R. and Pal S. (2021). Is meta embedding better than pre-trained word embedding to perform sentiment analysis for dravidian languages in code-mixed text?. In *FIRE*.
- Chanda S., Ujjwal S., Das S. and Pal S. (2021). Fine-tuning pre-trained transformer based model for hate speech and offensive content identification in english, indo-aryan and code-mixed (english-hindi) languages. In *Forum for Information Retrieval Evaluation (Working Notes)(FIRE)*, CEUR-WS. org.
- Gambäck B. and Das A. (2014). On measuring the complexity of code-mixing. In *Proceedings of the 11th International Conference on Natural Language Processing*, Goa, India, pp. 1–7.
- Guzman G.A., Ricard J., Serigos J., Bullock B.E. and Toribio A.J. (2017). Metrics for modeling code-switching across corpora. In *INTERSPEECH*, pp. 67–71.
- Hande A., Priyadharshini R. and Chakravarthi B. R. (2020). KanCMD: Kannada CodeMixed dataset for sentiment analysis and offensive language detection. In *Proceedings of the Third Workshop on Computational Modeling of People's Opinions, Personality, and Emotion's in Social Media, Association for Computational Linguistics*, Barcelona, Spain (Online), pp. 54–63.
- Hern A. (2021). WhatsApp loses millions of users after terms update. *The Guardian*, <https://www.theguardian.com/technology/2021/jan/24/whatsapp-loses-millions-of-users-after-terms-update>.
- Joshi A., Prabhu A., Shrivastava M. and Varma V. (2016). Towards sub-word level compositions for sentiment analysis of Hindi-English code mixed text. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, Osaka, Japan, pp. 2482–2491. The COLING 2016 Organizing Committee.
- Khanuja S., Dandapat S., Srinivasan A., Sitaram S. and Choudhury M. (2020). GLUECoS : An Evaluation Benchmark for Code-Switched NLP. arXiv: 2004.12376 [cs.CL].
- Kusampudi S.S.V., Sathineni P. and Mamidi R. (2021). Sentiment analysis in code-mixed Telugu-English text with unsupervised data normalization. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021)*, Held Online. INCOMA Ltd, pp. 753–760.
- Lee S. and Wang Z. (2015). Emotion in code-switching texts: Corpus construction and analysis. In *Proceedings of the Eighth SIGHAN Workshop on Chinese Language Processing*, Beijing, China: Association for Computational Linguistics, pp. 91–99.
- Mandal S. and Das D. (2018). Analyzing roles of classifiers and code-mixed factors for sentiment identification. *ArXiv*, [abs/1801.02581](https://arxiv.org/abs/1801.02581).
- Ortiz-Ospina E. (2019). “The rise of social media”. Published online at OurWorldInData.org. Retrieved from: <https://ourworldindata.org/rise-of-social-media> [Online Resource].
- Padmaja S., Fatima S., Bandu S., Nikitha M. and Prathyusha K. (2020). Sentiment extraction from bilingual code mixed social media text. In *Data Engineering and Communication Technology*. Springer, pp. 707–714.
- Pang B. and Lee L. (2004). A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, Barcelona, Spain, pp. 271–278. <https://doi.org/10.3115/1218955.1218990>. <https://www.aclweb.org/anthology/P04-1035>
- Patra B. G., Das D. and Das A. (2018). Sentiment analysis of code-mixed Indian languages. In *An Overview of SAIL Code-Mixed Shared Task @ICON-2017*.

- Patra B. G., Das D., Das A. and Prasath R.** (2015). Shared task on sentiment analysis in Indian languages sail Tweets - An overview. In *Proceedings of the Third International Conference on Mining Intelligence and Knowledge Exploration, MIKE 2015*, Berlin, Heidelberg: Springer-Verlag, vol 9468, pp. 650–655.
- Patwa P., Aguilar G., Kar S., Pandey S., S. P. Y. K. L., Gambäck B., Chakraborty T., Solorio T. and Das A.** (2020). Semeval-2020 task 9: Overview of sentiment analysis of code-mixed tweets.
- Pires T., Schlinger E. and Garrette D.** (2019). How multilingual is multilingual BERT? arXiv preprint arXiv: 1906.
- Priyadharshini R., Chakravarthi B. R., Thavareesan S., Chinnappa D., Durairaj T. and Sherly E.** (2021). Overview of the Dravidian codemix 2021 shared task on sentiment detection in Tamil, Malayalam, and Kannada. In *Forum for Information Retrieval Evaluation, FIRE 2021*. Association for Computing Machinery.
- Rijhwani S., Sequiera R., Choudhury M., Bali K. and Maddila C.S.** (2017). Estimating code-switching on Twitter with a novel generalized word-level language detection technique, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vancouver, Canada: Association for Computational Linguistics, pp. 1971–1982.
- Saroj A., Chanda S. and Pal S.** (2020). IRLab@IITV at SemEval-2020 task 12: Multilingual offensive language identification in social media using SVM. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, Barcelona (online): International Committee for Computational Linguistics, pp. 2012–2016.
- Sharma S., Srinivas P.Y.K.L. and Balabantaray R.C.** (2015). Text normalization of code mix and sentiment analysis. In *2015 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, IEEE, pp. 1468–1473.
- Sridhar S. N.** (1978). On the functions of code-mixing in Kannada. *International Journal of the Sociology of Language* 1978(16), 109–118.
- Thavareesan S. and Mahesan S.** (2019). Sentiment analysis in Tamil texts: A study on machine learning techniques and feature representation. In *2019 14th Conference on Industrial and Information Systems (ICIIS)*, pp. 320–325.
- Thavareesan S. and Mahesan S.** (2020a). Sentiment lexicon expansion using Word2vec and fastText for sentiment prediction in Tamil texts. In *2020 Moratuwa Engineering Research Conference (MERCon)*, pp. 272–276.
- Thavareesan S. and Mahesan S.** (2020b). Word embedding-based Part of Speech tagging in Tamil texts. In *2020 IEEE 15th International Conference on Industrial and Information Systems (ICIIS)*, pp. 478–482.
- Utsav J., Kabaria D., Vajpeyi R., Mina M. and Srivastava V.** (2020). Stance detection in Hindi-English code-mixed data, *Proceedings of the 7th ACM IKDD CoDS and 25th COMAD (CoDS COMAD 2020)*, New York, NY, USA: Association for Computing Machinery, pp. 359–360. <https://doi.org/10.1145/3371158.3371226>.
- Vilares D., Alonso M. A. and Gómez-Rodríguez C.** (2015). Sentiment analysis on monolingual, multilingual and code-switching Twitter corpora. In *Proceedings of the 6th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pp. 2–8.
- Vilares D., Alonso M. A. and Gómez-Rodríguez C.** (2016). EN-ES-CS: An English-Spanish code-switching Twitter corpus for multilingual sentiment analysis. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pp. 4149–4153.
- Vilares D., Alonso M. A. and Gómez-Rodríguez C.** (2017). Supervised sentiment analysis in multilingual environments. *Information Processing & Management* 53(3), 595–607.
- Wang Z., Lee S., Li S. and Zhou G.** (2015). Emotion detection in code-switching texts via bilingual and sentimental information. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pp. 763–768.
- Wang Z., Lee S. Y. M., Li S. and Zhou G.** (2016). Emotion analysis in code-switching text with joint factor graph model. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 25(3), 469–480.