# Is Meta Embedding better than pre-trained word embedding to perform Sentiment Analysis for Dravidian Languages in Code-Mixed Text?

Supriya Chanda[a], Rajat Pratap Singh[a] and Sukomal Pal[a]

[a]*Indian Institute of Technology (BHU), Varanasi, INDIA*

**Abstract**

This paper describes the IRlab@IITBHU system for the Dravidian-CodeMix - FIRE 2021: Sentiment Analysis for Dravidian Languages pairs Tamil-English (TA-EN), Kannada-English (KN-EN), and Malayalam-English (ML-EN) in Code-Mixed text. We have reported three models output in this paper where We have submitted only one model for sentiment analysis of all code-mixed datasets. Run-1 was obtained from the FastText embedding with multi-head attention, Run-2 used the meta embedding techniques, and Run-3 used the Multilingual BERT(mBERT) model for producing the results. Run-2 outperformed Run-1 and Run-3 for all the language pairs.

**Keywords**

Code Mixed, Kannada, Malayalam, Tamil, BERT, fastText, Sentiment Analysis,

## 1. Introduction

Web 2.0 has resulted in an exponential increase in the number of Web users and the volume of Web content. Mobile Internet and web users have made social media a massive industry in today's society; More than 75% of the world's population utilizes social media. The majority of these users are both consumers and producers of information. Recently, with the widespread adoption of social media platforms, the focus has shifted to code-mixed text. Text written in multiple languages makes up the code-mixed text. People naturally combine their native tongue with global languages such as English. Not only their native script, many times People use Roman script to express themselves in colloquial languages (transliteration).

Because these texts are primarily informal and casual, grammar rules are rarely followed. In the transliterated text, there is no established set of spelling rules. This liberation leads to large-scale spelling variations, which pose a significant challenge in processing mixed script data. Current NLP techniques are insufficient to process such texts. So that's why it became an essential research domain to deal with code mixed data.

Not only in academia, but the industry also has considerable interest for the last few years in many downstream tasks on code mixed data like Language Identification, POS tagging, NER,

*FIRE'21: Forum for Information Retrieval Evaluation, Dec 13–17, 2021*

supriyachanda.rs.cse18@itbhu.ac.in (S. Chanda); rajatp.singh.cd.che19@itbhu.ac.in (R.P. Singh); spal.cse@itbhu.ac.in (S. Pal)
https://cse-iitbhu.github.io/irlab/supriya.html (S. Chanda); https://cse-iitbhu.github.io/irlab/spal.html (S. Pal)
0000-0001-8743-9830 (S. Pal)

QA, NLI and Machine Translation. Understanding code-switched communication will help large corporations better target their advertising. Understanding genuine user feedback on product features aids in the development of future versions. Ignoring one language in favor of another or completely ignoring code-switched languages can lead to incorrect conclusions about user sentiment.

It's an essential study topic in natural language processing to understand how people feel about things. Code Mixed language writings have become increasingly frequent in media communication with the rise of social media. When analyzing a text, sentence, or paragraph, sentiment analysis is the act of determining the sentiments, such as emotions and affectionate to others.

GLUECoS [1] - an evaluation benchmark for code-mixed text - and LinCE [2] - a centralised benchmark for 10 corpora including four different code-switched language pairings and four tasks - have been conducted in this direction. In the past, code-switching workshops held in conjunction with major NLP conferences have included shared tasks. The first and second workshops on Computational Approaches to Code Switching shared a problem on Language Identification[1] for numerous language pairs (Nepalese-English, Spanish-English, Mandarin-English and Modern Standard Arabic-Arabic dialects). Another goal was to identify named entities[2] in the English-Spanish and Modern Standard Arabic-Egyptiac arabic language pairs, which was done in a shared task during the third workshop. Another shared assignment was Machine Translation[3] for many language combinations, which took place in the fourth workshop.

A number of code-switching tasks have been carried out by the Forum for Information Retrieval Evaluation (FIRE). Code-mixed entity extraction, POS tagging for code-mixed Indian social media (ICON 2016), sentiment analysis for code-mixed Indian languages [3] (ICON 2017), and the Code-Mixed Question Answering Challenge are just a few examples of the tasks. There was a competition for Sentiment Analysis in Code-Switched Data (*Task 9: Sentiment Analysis for Code-Mixed Social Media Text* [4]), which covered tweets in both Spanish-English and Hindi-English pairs.

The shared task [5] here aims to identify the sentiment polarity of the code-mixed data of YouTube comments in Dravidian Language pairs (Malayalam-English, Tamil-English, and Kannada-English) collected from social media. A new dataset has been included in this year's shared work for the second consecutive year. Like last year, we'll have to categorize the text into five different categories: Positive, Negative, Mixed_feelings, unknown_state and not-<*language*>[4]. To solve the above task, we clean the comments, construct a representation of comments with different word embedding methods, and then build the classification model. All of the models' test data findings are included in this report.

The rest of the paper is organized as follows. Section 2 describes the dataset, pre-processing and processing techniques. Model architecture are described in Section 3. In Section 4, we report our results and analysis. Finally we conclude in Section 5.

[1] http://emnlp2014.org/workshops/CodeSwitch/call.html

[2] https://code-switching.github.io/2018/#shared-task-id

[3] https://code-switching.github.io/2021

[4] The language might be Tamil, Kannada or Malayalam

**Table 1**
Data Distribution

| 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 |

# 2. System Description

## 2.1. Datasets

The Dravidian-CodeMix shared task[5] organizers provided a dataset for Training, Development and test. The training dataset consists of 35,656 Tamil-English [6] and 6,212 Kannada-English [7] and 15,880 Malayalam-English [8] YouTube video comments. The statistics of training, development, and test data corpus collection and their class distribution are shown in Table 1. The details of the dataset and benchmark results are given in overview [9] and findings [10] of the Sentiment Analysis of Dravidian Languages. The dataset provided suffers from general problems of social media data, particularly code-mixed data. The sentences are short with lack of well-defined grammatical structures, and many spelling mistakes.

[5] https://dravidian-codemix.github.io/2021/index.html

## 2.2. Data Pre-processing

The YouTube comment dataset used in this work is already labelled into five categories: Positive, Negative, Mixed_feelings, unknown_state and not-<*language*>[6]. Our pre-processing of comments includes the following steps:

- In the previously shared task report [11], we have seen that removing contiguous repeating characters does not give any significant performance changes. That's why this year, we didn't perform any removal of adjacent repeating characters.
- Removal of exclamations and other punctuation
- Removal of non-ASCII characters, all the emoticons, symbols, numbers, special characters.

## 2.3. Word Embedding

Word embedding is arguably the most widely known technology in the recent history of NLP. It captures the semantic property of a word. We use `bert-base-multilingual-cased` pre-trained models[7], `FastText` [12] and `TF-iDF` [13] to get a vector as an embedding for the sentence that we can use for classification.

- **fastText:** fastText, developed by Facebook, combines certain concepts introduced by the NLP and ML communities, representing sentences with a bag-of-words and n-grams using subword information and sharing them across classes through a hidden representation. fastText[14] can learn vector representations of out-of-vocabulary words, which is useful for our dataset that contains Malayalam and Tamil words in Roman script.
- **mBERT**: A transformer architecture is an encoder-decoder network that uses self-attention on the encoder side and attention on the decoder side. The models are pre-trained 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 (Bidirectional Encoder Representations from Transformers) has been used for the downstream task of code-mixed language identification. Multilingual BERT or mBERT (`bert-base-multilingual-cased`[8]) is pre-trained on cased text in the top 104 languages with the largest wikipedias and has a total 179M 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, passes the result through a feedforward network to the next encoder.

[6] The language might be Tamil, Kannada or Malayalam

[7] https://huggingface.co/transformers/pretrained_models.html

[8] https://github.com/google-research/bert/blob/master/multilingual.md

## 3. Model Architecture

In this section, we summarise the modules that make up our model. The text input is first tokenized using a language-independent subword tokenizer and SentencePiece. It performs the subword segmentation supporting the byte-pair-encoding (BPE) algorithm and unigram language model. Then it converts this text into an id sequence to guarantee perfect reproducibility of the normalization and subword segmentation. The proposed model uses the fastText embeddings to represent the vectors for the tokenized text as input. The main objective of the fastText embeddings is to consider the internal structure of words instead of learning word representations. Because of this, morphologically rich languages can learn their word representations independently. Instead of learning vectors for words directly, fastText represents each word as an n-gram of characters. This ensures that the words love, loved, and beloved all have similar vector representations, even if they appear in different contexts. This feature enhances learning on heavily inflected languages. A skip-gram model is trained to understand the embeddings once the word has been represented using character n-grams.

Attention-based models have been used in various topics, including sentiment analysis [15]. In [16], the author has devised an architecture that improves and performs well beyond the baseline using the Multi-Head attention mechanism. Moving in the same direction, we use a Multi-Head Attention-based transformer encoder to get attention-aware context vectors for the sentences. We add positional encoding to the word embedding vector before the first self-attention layer to retain the notion of order. Self-attention enables us to find correlations between different input words, indicating the syntactic and contextual structure of the sentence. The encoded vectors now having efficacy on word-level are then passed from a bi-LSTM layer. A classifier layer is used to predict the sentiment label of the input based on the output hidden representations of the bi-LSTM layer.

In our revised approach, we formulate meta-embedding by concatenating tf-idf vectors of the tokenized texts and the hidden representations of the bi-LSTM layer. TF-IDF gives us a way to associate each word in a document with a number that represents how relevant each word is in that document. Then, documents with similar, appropriate words will have similar vectors. The Meta embeddings form a more semantically and syntactically effective representation of the input text, thus improving the score significantly. For the hyper-parameters,we considered 5 training rounds, a batch size of 16, learning rate of 5e-4 along with a dropout value of 0.1. fastText embedding of 300D are trained over 15 iterations and we built one bi-LSTM layer of hidden dimension size 256.

After evaluating the mBERT model on validation data, the hyper-parameter were set. We have used the following hyper-parameters: `batch size` = 32, `learning rate` = 2e-5, optimizer = AdamW, epochs = 4.

## 4. Results and Analysis

In our submission, we considered a significantly large number of epochs as compared to the updated version, from 15 to 5 which resulted in the model over-fitting for the task. We also considered to ignore the PAD_IDX for Cross Entropy Loss which resulted in the model not

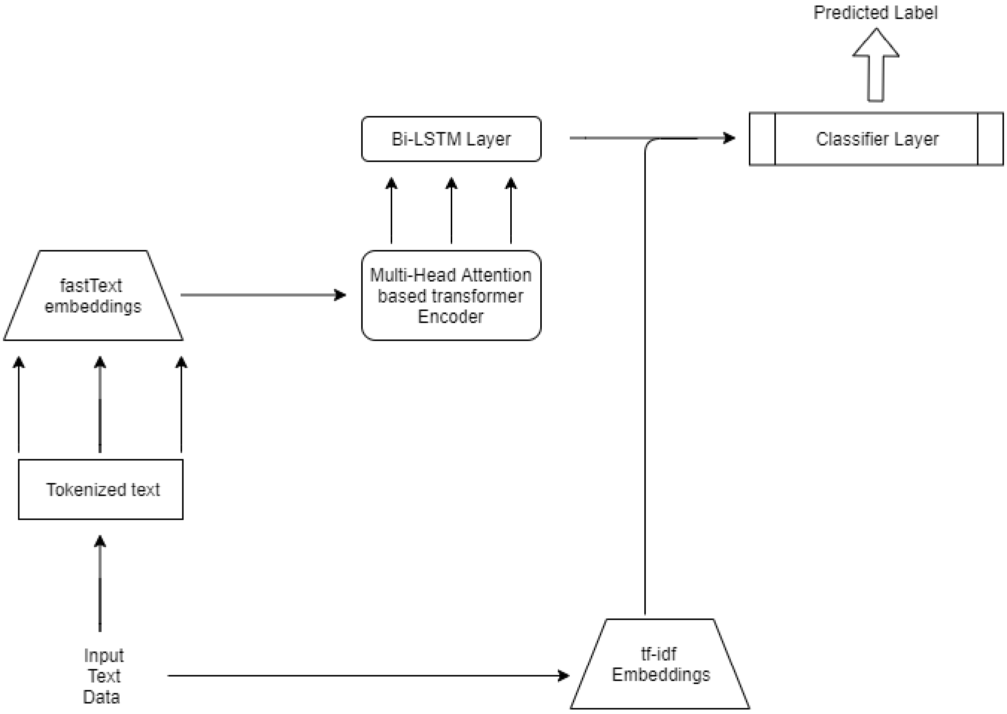


**Figure 1:** Model Architecture for Meta embedding

converging and failing to predict certain labels at all. In the updated approach, we also used only one layer for bi-LSTM network to avoid over-fitting. As seen and evident from the results, the model and the proposed architecture performs significantly better producing competitive scores for the task of sentiment analysis of code-mixed data.

Multilingual BERT based experiments was performed on Google's Colab[9]. PyTorch deep learning library has been used to implement the models. We also use HuggingFace's transformers to fine-tune pre-trained mBERT models. A Macro $F_1$ score was used to evaluate every system. Macro $F_1$ score of the overall system was the average of $F_1$ scores of the individual classes. Table 2, Table 3 and Table 4 shows our official and unofficial performances as shared by the organizers vis-a-vis the best performing team for Tamil, Kananda and Malayalam language pair respectively. Table 5, Table 6 and Table 7 report the individual classwise Precision, Recall and $F_1$ score on Tamil-English, Kannada-English and Malayalam-English corpus respectively.

We used the confusion matrix for additional analysis (See Fig. 2). When describing the performance of a classification model on test data for which the true values are known, confusion matrix tables are often used.

we could not verify the accuracy of the labelling because we do not understand Tamil, Kannada or Malayalam. All model performed well in positive class follow by not_<language> class. The reason behind this is the imbalanced data in corpus. For our first run that we have submit-

[9] https://colab.research.google.com

**Table 2**
Evaluation results on Tamil-English test data and rank list

| | **Tamil - English** | | | |
|---|---|---|---|---|
| Team Name | Precision | Recall | $F_1$ score | Rank |
| CIA_NITT | 0.709 | 0.714 | 0.711 | 1/22 |
| IRLab@IITBHU (Run-1) | 0.375 | 0.457 | 0.412 | 20/22 |
| IRLab@IITBHU (Run-2: Not Submitted) | 0.62 | 0.64 | **0.63** | - |
| IRLab@IITBHU (Run-3: Not Submitted) | 0.61 | 0.63 | 0.61 | - |

**Table 3**
Evaluation results on Kannada-English test data and rank list

| | **Kannada - English** | | | |
|---|---|---|---|---|
| Team Name | Precision | Recall | $F_1$ score | Rank |
| SSNCSE_NLP | 0.639 | 0.656 | 0.63 | 1/15 |
| IRLab@IITBHU (Run-1) | 0.291 | 0.35 | 0.317 | 15/15 |
| IRLab@IITBHU (Run-2: Not Submitted) | 0.63 | 0.66 | **0.64** | - |
| IRLab@IITBHU (Run-3: Not Submitted) | 0.60 | 0.64 | 0.61 | - |

**Table 4**
Evaluation results on Malayalam-English test data and rank list

| | **Malayalam - English** | | | |
|---|---|---|---|---|
| Team Name | Precision | Recall | $F_1$ score | Rank |
| ZYBank-AI Team | 0.803 | 0.806 | 0.804 | 1/15 |
| IRLab@IITBHU (Run-1) | 0.648 | 0.665 | 0.653 | 10/15 |
| IRLab@IITBHU (Run-2: Not Submitted) | 0.72 | 0.70 | **0.71** | - |
| IRLab@IITBHU (Run-3: Not Submitted) | 0.69 | 0.70 | 0.69 | - |

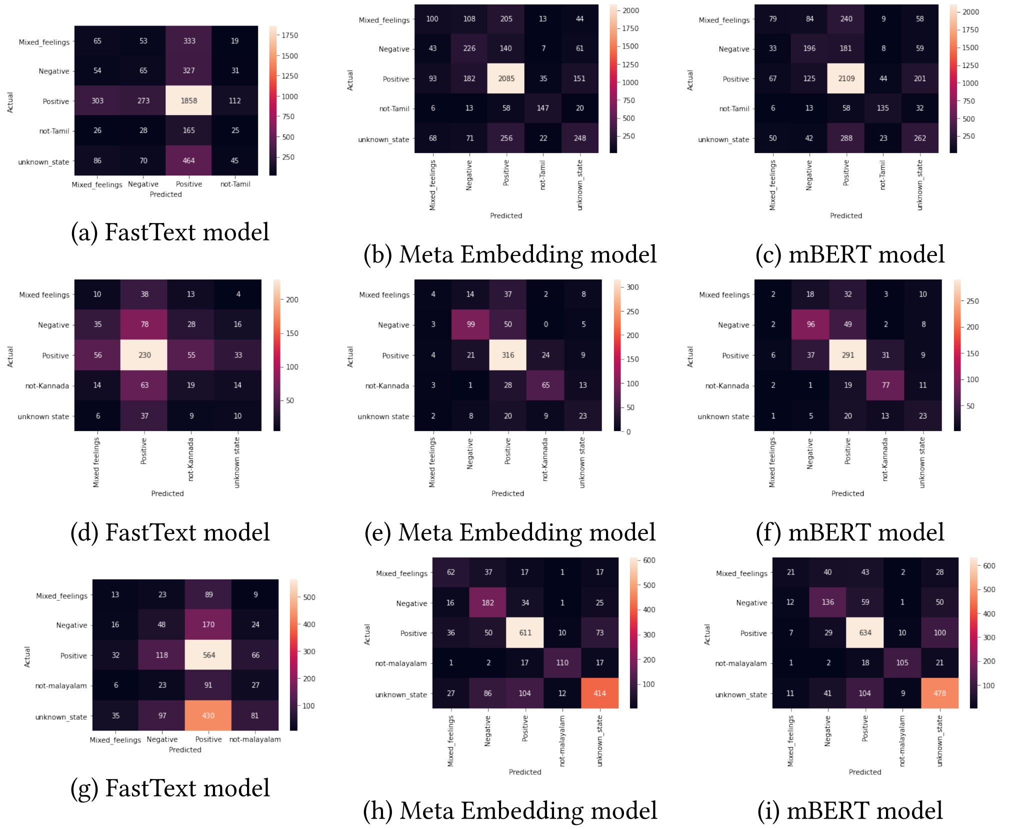
**Table 5**
Precision, recall, $F_1$-score, and support for all experiment on Tamil-English test data

| | **mBERT** | | | **Meta Embedding** | | | **FastText** | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Precision | Recall | $F_1$-score | Precision | Recall | $F_1$-score | Precision | Recall | $F_1$-score | support |
| Mixed_feelings | 0.34 | 0.17 | 0.22 | 0.32 | 0.21 | 0.26 | 0.12 | 0.14 | 0.13 | 470 |
| Negative | 0.43 | 0.41 | 0.42 | 0.38 | 0.47 | 0.42 | 0.13 | 0.14 | 0.13 | 477 |
| Positive | 0.73 | 0.83 | 0.78 | 0.76 | 0.82 | 0.79 | 0.59 | 0.73 | 0.65 | 2546 |
| not-Tamil | 0.62 | 0.55 | 0.58 | 0.66 | 0.60 | 0.63 | 0.11 | 0.10 | 0.11 | 244 |
| unknown_state | 0.43 | 0.39 | 0.41 | 0.47 | 0.37 | 0.42 | 0.00 | 0.00 | 0.00 | 665 |
| macro avg | 0.51 | 0.47 | 0.48 | 0.52 | 0.50 | 0.50 | 0.19 | 0.22 | 0.20 | 4402 |
| weighted avg | 0.61 | 0.63 | 0.61 | 0.62 | 0.64 | 0.63 | 0.37 | 0.46 | 0.41 | 4402 |
| Accuracy | | 0.63 | | | 0.64 | | | 0.46 | | |

ted for evaluation, the model cannot classify unknown_state class on both Tamil-English and Malayalam-English dataset and missed Negative class on Kannada-English dataset.

## 5. Conclusion

This study reports performance of our system for the shared task on Sentiment Analysis for Dravidian Languages in Code-Mixed Text in Dravidian-CodeMix - FIRE 2021. We conducted a



(a) FastText model

(b) Meta Embedding model

(c) mBERT model

(d) FastText model

(e) Meta Embedding model

(f) mBERT model

(g) FastText model

(h) Meta Embedding model

(i) mBERT model

**Figure 2:** Confusion matrices for all proposed models on the Corpus Test Set. (a) Run-1 on TA-EN, (b) Run-2 on TA-EN, (c) Run-3 on TA-EN, (d) Run-1 on KN-EN, (e) Run-2 on KN-EN, (f) Run-3 on KN-EN, (g) Run-1 on ML-EN, (h) Run-2 on ML-EN, (i) Run-3 on ML-EN

**Table 6**

Precision, recall, $F_1$-scores, and support for all experiment on Kannada-English test data

| | **mBERT** | | | **Meta Embedding** | | | **FastText** | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Precision | Recall | $F_1$-score | Precision | Recall | $F_1$-score | Precision | Recall | $F_1$-score | support |
| Mixed_feelings | 0.15 | 0.03 | 0.05 | 0.25 | 0.06 | 0.10 | 0.08 | 0.15 | 0.11 | 65 |
| Negative | 0.61 | 0.61 | 0.61 | 0.69 | 0.63 | 0.66 | 0.00 | 0.00 | 0.00 | 157 |
| Positive | 0.71 | 0.78 | 0.74 | 0.70 | 0.84 | 0.77 | 0.52 | 0.61 | 0.56 | 374 |
| not-Kannada | 0.61 | 0.70 | 0.65 | 0.65 | 0.59 | 0.62 | 0.15 | 0.17 | 0.16 | 110 |
| unknown_state | 0.38 | 0.37 | 0.37 | 0.40 | 0.37 | 0.38 | 0.13 | 0.16 | 0.14 | 62 |
| macro avg | 0.49 | 0.50 | 0.49 | 0.54 | 0.50 | 0.51 | 0.18 | 0.22 | 0.19 | 768 |
| weighted avg | 0.60 | 0.64 | 0.61 | 0.63 | 0.66 | 0.64 | 0.29 | 0.35 | 0.32 | 768 |
| Accuracy | 0.64 | | | 0.66 | | | 0.35 | | | |

number of experiments on a real-world code-mixed YouTube comments dataset involving a few embedding techniques: fastText, Multilingual BERT, and Tf-idf. We find that Meta embedding model outperforms pre-trained word embedding like mBERT on this task. There's still scope of improvement for the labels classified as not_Language as we suggest to add word language specific embedding to the text vectors and can consider several other methods for the future

**Table 7**
Precision, recall, $F_1$-scores, and support for all experiment on Malayalam-English test data

| | **mBERT** | | | **Meta Embedding** | | | **FastText** | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Precision | Recall | $F_1$-score | Precision | Recall | $F_1$-score | Precision | Recall | $F_1$-score | support |
| Mixed_feelings | 0.40 | 0.16 | 0.23 | 0.44 | 0.46 | 0.45 | 0.13 | 0.10 | 0.11 | 134 |
| Negative | 0.55 | 0.53 | 0.54 | 0.51 | 0.71 | 0.59 | 0.16 | 0.19 | 0.17 | 258 |
| Positive | 0.74 | 0.81 | 0.77 | 0.78 | 0.78 | 0.78 | 0.42 | 0.72 | 0.53 | 780 |
| not-malayalam | 0.83 | 0.71 | 0.77 | 0.82 | 0.75 | 0.78 | 0.13 | 0.18 | 0.15 | 147 |
| unknown_state | 0.71 | 0.74 | 0.72 | 0.76 | 0.64 | 0.70 | 0.00 | 0.00 | 0.00 | 643 |
| macro avg | 0.64 | 0.59 | 0.61 | 0.66 | 0.67 | 0.66 | 0.17 | 0.24 | 0.19 | 1962 |
| weighted avg | 0.69 | 0.70 | 0.69 | 0.72 | 0.70 | 0.71 | 0.21 | 0.33 | 0.25 | 1962 |
| Accuracy | 0.70 | | | 0.70 | | | 0.33 | | | |

work.

## References

[1] S. Khanuja, S. Dandapat, A. Srinivasan, S. Sitaram, M. Choudhury, Gluecos : An evaluation benchmark for code-switched nlp, 2020. arXiv:2004.12376.

[2] G. Aguilar, S. Kar, T. Solorio, LinCE: A Centralized Benchmark for Linguistic Code-switching Evaluation, in: Proceedings of The 12th Language Resources and Evaluation Conference, European Language Resources Association, Marseille, France, 2020, pp. 1803–1813. URL: https://www.aclweb.org/anthology/2020.lrec-1.223.

[3] B. G. Patra, D. Das, A. Das, Sentiment Analysis of Code-Mixed Indian Languages: An Overview of SAIL Code-Mixed Shared Task @ICON-2017, 2018. arXiv:1803.06745.

[4] P. Patwa, G. Aguilar, S. Kar, S. Pandey, S. PYKL, B. Gambäck, T. Chakraborty, T. Solorio, A. Das, Semeval-2020 task 9: Overview of sentiment analysis of code-mixed tweets, 2020. arXiv:2008.04277.

[5] B. R. Chakravarthi, R. Priyadharshini, V. Muralidaran, S. Suryawanshi, N. Jose, J. P. Sherly, Elizabeth McCrae, Overview of the track on Sentiment Analysis for Davidian Languages in Code-Mixed Text, in: Proceedings of the 12th Forum for Information Retrieval Evaluation, FIRE '20, 2020.

[6] B. R. Chakravarthi, V. Muralidaran, R. Priyadharshini, J. P. McCrae, 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, 2020, pp. 202–210. URL: https://www.aclweb.org/anthology/2020.sltu-1.28.

[7] A. Hande, R. Priyadharshini, B. R. Chakravarthi, 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), 2020, pp. 54–63. URL: https://www.aclweb.org/anthology/2020.peoples-1.6.

[8] B. R. Chakravarthi, N. Jose, S. Suryawanshi, E. Sherly, J. P. McCrae, 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, 2020, pp. 177–184. URL: https://www.aclweb.org/anthology/2020.sltu-1.25.

[9] R. Priyadharshini, B. R. Chakravarthi, S. Thavareesan, D. Chinnappa, T. Durairaj, E. Sherly, Overview of the dravidiancodemix 2021 shared task on sentiment detection in tamil, malayalam, and kannada, in: Forum for Information Retrieval Evaluation, FIRE 2021, Association for Computing Machinery, 2021.

[10] B. R. Chakravarthi, R. Priyadharshini, S. Thavareesan, D. Chinnappa, D. Thenmozhi, E. Sherly, J. P. McCrae, A. Hande, R. Ponnusamy, S. Banerjee, C. Vasantharajan, Findings of the Sentiment Analysis of Dravidian Languages in Code-Mixed Text, in: Working Notes of FIRE 2021 - Forum for Information Retrieval Evaluation, CEUR, 2021.

[11] S. Chanda, S. Pal, Irlab@ iitbhu@ dravidian-codemix-fire2020: Sentiment analysis for dravidian languages in code-mixed text (2020).

[12] P. Bojanowski, E. Grave, A. Joulin, T. Mikolov, Enriching word vectors with subword information, Transactions of the Association for Computational Linguistics 5 (2017) 135–146. URL: https://aclanthology.org/Q17-1010. doi:10.1162/tacl_a_00051.

[13] A. Aizawa, An information-theoretic perspective of tf–idf measures, Information Processing Management 39 (2003) 45–65. URL: https://www.sciencedirect.com/science/article/pii/S0306457302000213. doi:https://doi.org/10.1016/S0306-4573(02)00021-3.

[14] T. Mikolov, E. Grave, P. Bojanowski, C. Puhrsch, A. Joulin, Advances in Pre-Training Distributed Word Representations, in: Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018), 2018.

[15] Y. Wang, M. Huang, X. Zhu, L. Zhao, Attention-based LSTM for aspect-level sentiment classification, in: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Austin, Texas, 2016, pp. 606–615. URL: https://aclanthology.org/D16-1058. doi:10.18653/v1/D16-1058.

[16] X. Zhang, T. Gao, Multi-head attention model for aspect level sentiment analysis, 2020. doi:10.3233/JIFS-179383.