CoMaTa OLI-Code-mixed Malayalam and Tamil Offensive Language Identification

F. Balouchzahi[1], S. Bashang[2], G. Sidorov[1] and H. L. Shashirekha[3]

[1]Instituto Politécnico Nacional (IPN), Centro de Investigación en Computación (CIC), Mexico City, Mexico
[2]Canara Bank School of Management Studies, Bangalore University, India
[3]Department of Computer Science, Mangalore University, Mangalore, India

Abstract

Offensive Language Identification (OLI) in code-mixed under-resourced Dravidian languages is a challenging task due to the complex characteristics of code-mixed text and scarcity of digital resources and tools to process these languages. This paper describes the strategy proposed by our team MUCIC for the 'Dravidian-CodeMix-HASOC2021' shared task which includes two tasks: Task 1 and Task 2, with the aim of classifying a given social media post/comment into one of two predefined categories: Offensive (OFF) and Not-Offensive (NOT) in both the tasks. While Task 1 aims at identifying Hate Speech (HS) contents in Tamil language in native script, Task 2 focuses on identifying HS contents in Tamil-English (Ta-En) and Malayalam-English (Ma-En) code-mixed texts in Roman script. Training the Machine Learning (ML) classifiers using the most frequent char and word n-grams, the proposed methodology secured 2nd, 1st, and 2nd ranks for Tamil, and Ta-En and Ma-En code-mixed texts with average weighted F1-scores of 0.852, 0.678, and 0.762 respectively.

Keywords

Code-mixed, HASOC, Dravidian languages, n-grams, Machine Learning

1. Introduction

The current era is witnessing a tremendous increase in the use of social media platforms all over the globe. Covid-19 situation has increased this further since a year due to lockdown and isolations [1]. Due to quick and easy access, people use social media as a mode of communication to interact, connect, and express their thoughts and opinions about various things like a movie, situation like Covid-19, the present situation in Afghanistan, and so on. It is observed that, in multilingual societies like India, users may post their comments mixing two or more languages in a sentence, word or sub-word which leads to generating a large amount of code-mixed content [2, 3].

Code-mixing plays a vital role in maximizing the communication effectively between social media users. According to Pfaff et al. [4], the bilingual competence of individuals results in producing more code-mixed content on social media. Code-mixing can also be considered as an

FIRE 2021, Forum for Information Retrieval Evaluation, December 13-17, 2021, India
frs_b@yahoo.com (F. Balouchzahi); Sepidehbashang92@gmail.com (S. Bashang); sidorov@cic.ipn.mx (G. Sidorov); hlsrekha@gmail.com (H. L. Shashirekha)
https://mangaloreuniversity.ac.in/dr-h-l-shashirekha (H. L. Shashirekha)
0000-0003-1937-3475 (F. Balouchzahi)

asset that helps the bilingual/native speakers to convey certain meanings, special attitudes, or emotions effectively, especially in a multilingual scenario where there are no equivalent lexical items and cannot be translated into a second language.

Some reasons for code-mixing in the Indian context are as follows:

- Restricted vocabulary of individuals;
- Habitual expression;
- To convey special attitude or special meaning;
- Mood of the speaker;
- Identity marker;
- Profession;
- To show respect and relation;
- To criticize.

Models developed to process and analyse any monolingual data in general, may not give good performance on code-mixed data due to the complexity and lack of standardization of code-mixed data. Hence, research on code-mixed data of different levels of complexity should be encouraged significantly for different applications like Sentiments Analysis (SA), HS detection, and Offensive Language Identification (OLI). This, in turn, will contribute towards developing better models and approaches to solve complex problems involving code-mixed data.

The offensive content is a common impolite phenomenon in real life and has negative impacts on individuals and societies. In fact, a person or a group of people can be targeted with offensive language due to the same ethnicity, gender or sexual orientation, political affiliation, religious belief, or any other characteristics [5]. Among the code-mixed content on social media, offensive content targeting an individual or a group is increasing day-by-day. OLI task demands automated methods as filtering such content manually is labor intensive and error prone.

Kannada, Malayalam, and Tamil are the Dravidian languages spoken in Karnataka, Kerala and Tamil Nadu states of India respectively. Despite their popularity these languages are under-resourced due to the scarcity of digital resources and tools for processing these languages [6]. Further, the amount of work done on the Natural Language Processing (NLP) tasks of these languages is very much limited [7, 8]. Usually, the native speakers of these Dravidian languages mix English with their languages at different linguistic units such as sentence, word, morpheme and sub-word to post comments and share information on social media which leads to the generation of code-mixed data. The nature of code-mixed data and the under-resourcefulness of Dravidian languages makes the OLI task more complex and challenging [9].

OLI in Dravidian languages[1] [10] is one of the shared tasks in FIRE 2021[2] which consists of two tasks, Task 1 and Task 2, with the aim of classifying a comment/post into Offensive (OFF) or NOT-Offensive (NOT) categories. Task 1 includes classifying comments written in Tamil language in native script and contains an extra category for text written in other languages that should be classified into not-Tamil, whereas Task 2 includes classifying comments in Ta-En and Ma-En code-mixed texts written in Roman script.

[1] https://competitions.codalab.org/competitions/31146
[2] http://fire.irsi.res.in/fire/2021/home

The literature and the findings of OLI shared task in DravidianLangTech-2021 [7] - the first workshop on Speech and Language Technologies for Dravidian Languages[3], reveals that most of the successful models in the shared task employed multilingual transformers. However, few traditional Machine Learning (ML) models submitted to this task also performed well but used a combination of complicated features extracted from texts instead of using any feature selection/ reduction algorithms. Using ML models with feature selection algorithms keeps the model simple compared to using a transformer like BERT which is resource intensive.

With the aim of exploring the neglected ML models, in this paper, we, team MUCIC, describe the models submitted to the OLI shared task in FIRE 2021. In this proposed work, we extend our previous work [9] by combining the most frequent word and char n-grams features extracted from the text and used them to train the three individual ML classifiers as well as an ensemble of these three ML classifiers. The results illustrate that the ML classifiers outperformed most of the models submitted to the OLI shared task and obtained 2nd rank in Task 1 and 1st and 2nd ranks in Task 2 for code-mixed Ta-En and Ma-En language pairs respectively. Further, team MUCIC emerged as the best performing team in the shared task w.r.t the average of the scores of the two tasks. The code for the proposed method is publicly available in our GitHub link[4].

The rest of the paper is organized as follows: Section 2 describes the related work followed by the proposed methodology in Section 3. Experiments and results are described in Section 4 and the paper concludes in Section 5.

2. Related Work

Research related to OLI in code-mixed texts is gaining lot of importance. Developing annotated datasets is an initial and major step in promoting various NLP applications for any natural or code-mixed language text. In this direction, Ta-En [11], Ka-En [12], and Ma-En[13] code-mixed datasets have been developed for SA/OLI tasks. These datasets were provided to the participants of the concerned shared tasks to develop and submit the working models for final evaluation and ranking.

OLI shared task in DravidianLangTech-2021 is the first workshop on Speech and Language Technologies for Dravidian languages [7, 8]. Several models were submitted by the researchers to this shared task and the top performing models are described below:

Inspired by the work of Zampieri et al. [14], Chakravarthi et al. [7] organized a shared task on OLI in code-mixed Dravidian languages, namely: Tamil, Malayalam and Kannada at various degrees of complexity. Datasets for this work were collected from different YouTube comments posted on movie trailers of Tamil, Kannada, and Malayalam in 2019. The authors adopted the Comment Scraper tool[5] as well as Langdetect[6] library to distinguish the languages used in the comments. Collected comments were classified into one of the six categories:

1. Not Offensive (comment does not mean to offend a person or group)
2. Offensive Untargeted (comment contains offensive words, but not directed at anyone)

[3]https://dravidianlangtech.github.io/2021/
[4]https://github.com/fazlfrs/CoMaTa-OLI
[5]https://github.com/philbot9/youtube-remarkscraper
[6]https://pypi.org/venture/langdetect/

3. Offensive Targeted Individual (comment targets a person by offensive words)
4. Offensive Targeted Group (comment contain offensive words targeting a group)
5. Offensive Targeted Other (comment contains offensive words but the targeted entity is not clear)
6. Not in intended language (comment written in languages other than the intended language)

In addition to the categories defined by Zampieri et al. [14], a new category "Not in intended language" was added to incorporate remarks written in languages other than the intended language. For instance, in the code-mixed Malayalam dataset, if the comment does not include Malayalam words written either in Malayalam or Roman script, it is considered as 'Not in intended language'. Participating teams were ranked based on the average weighted F1-scores of the predictions on the Test set.

Multilingual transformers such as Multilingual BERT[7] (mBERT), XLM-Roberta[8], and IndicBERT[9] were widely used in the models submitted by the participants of the shared task and some of the best performing models are briefly described below:

Saha et al. [15] fine-tuned various multilingual transformers including XLM-Roberta, mBERT, IndicBERT, and MuRIL[10] individually using unweighted and weighted cross-entropy loss functions for the shared task. For training, they used HuggingFace[11] with PyTorch[12] and the Adam adaptive optimizer with an initial learning rate of 1e-5. They also proposed a new BERT-Convolutional Neural Network (CNN) fusion classifier where they trained a single classifier on the concatenated embeddings from BERT and CNN models. For the CNN model, they used the 128-dim final layer embeddings trained on Skipgram word vectors and for the fusion classifier, a feed-forward neural network having four layers with batch normalization and dropout on the final layer was used. Further, they used Genetic Algorithm (GA)-optimized weighted ensembling and obtained average weighted F1-scores of 0.78 (1^{st} rank) for Ta-En, 0.74 (2^{nd} rank) for Ka-En, and 0.97 (1^{st} rank) for Ma-En language pairs.

Jayanthi et al. [16] ensembled mBERT and XLM-Roberta models which were pre-trained with their respective corpora utilizing Masked Language Modeling (MLM) objective. They transliterated the given datasets and used the HuggingFace library for the implementation of their backbone models. Further, they also proposed a fusion-architecture to leverage character-level, subword-level, and word-level embedding to boost the performance of their models. Their ensembled model secured the 1^{st} rank for Ka-En, 2^{nd} fpr Ma-En, and 3^{rd} for Ta-En languages pairs with 0.75, 0.97, and 0.76 average weighted F1-scores respectively.

Vasantharajan et al. [17] fine-tuned the mBERT transformer for the task of OLI in Dravidian languages. Primarily, in data pre-processing step, Emojis were converted into English text using a dictionary of Emojis and punctuation were removed from the texts. They employed BERT Tokenizer which converts the word into tokens and generates token ids, input masks (to increase the performance), and input type ids according to the input word. In addition, they also

[7] https://github.com/google-research/bert/blob/master/multilingual.md
[8] https://huggingface.co/transformers/model_doc/xlmroberta.html
[9] https://huggingface.co/ai4bharat/indic-bert
[10] https://huggingface.co/google/muril-base-cased
[11] https://huggingface.co/
[12] https://pytorch.org/

added a dropout and pooled output layer and obtained 2nd, 5th, and 6th ranks for Ma-En, Ka-En, and Ta-En language pairs with 0.96, 0.70, and 0.73 average weighted F1-scores respectively.

In their proposed model, Li et al. [18] removed Emojis and extra blanks in the code-mixed data to enhance the performance of the model and used class combination, class weights, and focal loss to allay the class-imbalance issue that existed in the training data. Finally, employing adversarial training to fine-tune XLM-Roberta and mBERT models, they achieved average weighted F1- scores of 0.75, 0.94, and 0.72 with ensembling the fine-tuned models for Ta-En, Ma-En, and Ka-En language pairs respectively.

Along with transformers-based models, few ML-based models using n-grams of various levels also performed well in the shared task. Few top performing ML based models are described below:

Balouchzahi et al. [9] proposed two models, namely, COOLI- Ensemble and COOLI-Keras. COOLI-Ensemble is a Voting Classifier (VC) with three ML estimators, namely: Multi-Layer Perceptron (MLP), eXtreme Gradient Boosting (XGB), and Logistic Regression (LR) and COOLI-Keras is based on Deep Learning (DL) approach. Pre-processing includes de-emojizing (converting Emoji to English text), removing punctuation, unnecessary characters and words of length less than 2, and converting English words to lower case. A set of char sequences and words are extracted as features and transformed to TFIDF vectors to train the learning models. COOLI-Ensemble model outperformed the COOLI-Keras model for both Ma-En and Ta-En language pairs and achieved 1st rank for Ma-En language pair with 0.97 average weighted F1-score and 4th and 6th ranks with 0.75 and 0.69 average weighted F1-scores for Ta-En and Kn-En language pairs respectively.

Bharathi et al. [19] vectorized word n-grams using CountVectorizer and TfidfVectorizer and concatenated them with mBERT embeddings vectors to train different ML classifiers, namely: MLP, k-Nearest Neighbor (kNN), Support Vector Machine (SVM), Random Forest (RF), and Decision Tree (DT) using these vectors. The best performances reported by the authors are the average weighted F1-scores of 0.73, 0.95, and 0.70 for Ta-En, Ma-En, and Kn-En language pairs respectively. Bhargav et al. [20] fed character n-grams vectors obtained by TfidfVectorizer for Ta-En and Ka-En language pairs to LR classifiers and obtained vectors from MuRIL language model for Ma-En texts to train a RF classifier. They obtained average weighted F1-scores of 0.95, 0.71, and 0.65 for Ma-En, Ta-En, and Ka-En language pairs respectively.

Despite the several learning models and several features, none of the models promise 100% accurate results which gives scope to explore further.

3. Methodology

The proposed strategy contains two main modules, namely: feature engineering and model construction, which are detailed below:

- **Feature Engineering:** This module consists of pre-processing texts by converting Emoji's to English words followed by removing punctuation, unnecessary characters and words of length less than 2 and lower casing the English text. TfidfVectorizer from Sklearn library is used to extract word n-grams in the range (1, 3) and char n-grams in the range (2, 5) from the pre-processed text, limiting the features to 40,000 frequent features in each

Table 1
Total number of features

Tasks	Language	Total number of n-grams	
		Char n-grams in the range (1, 3)	Word n-grams in the range (2, 5)
Task 1	Tamil	159,597	104,816
Task 2	Ta-En	99,110	107,951
	Ma-En	103,354	83,136

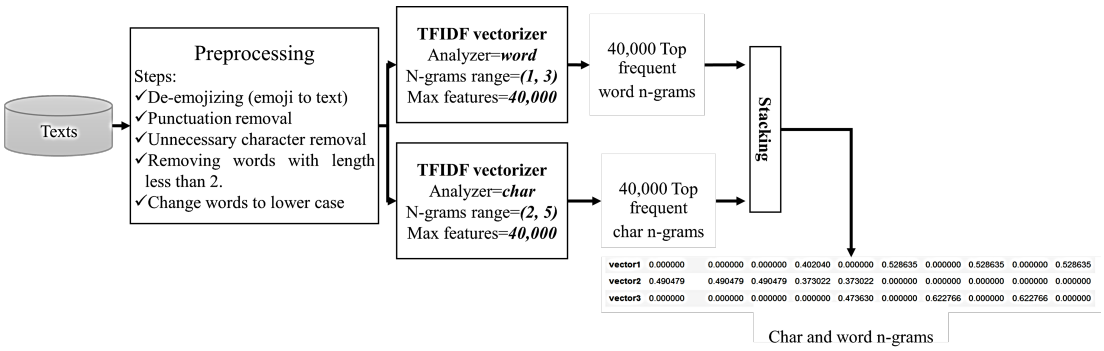


Figure 1: Feature engineering

case. These 40,000 frequent features of each type are stacked to obtain 80,000 features' feature vector. The total number of features without any limitation are presented in Table 1. Employing only the top frequent features reduces the dimensions and hence reduces the time taken to train the classifiers. In addition, it is expected that training the classifiers with only frequent features avoids overfitting and leads to enhancing the performance of classifiers. Figure 1 presents the graphical representation of feature engineering module.

- **Model Construction:** The three ML classifiers, namely: Linear SVM (LSVM), LR, and Random Forest (RF), and their ensemble with soft voting as shown in Figure 2 are trained using the feature vectors obtained for the training set. In soft voting ensemble model, the prediction of estimators which are the probability values for classes will be weighted by the classifier's importance and the largest sum of the weighted probabilities determine the final tag. The models are evaluated locally using the feature vectors obtained for the Development set and the predictions on the Test set are submitted to the organizers for final evaluation and ranking.

Evidently, it is expected that finding the best range of char and word n-grams and selecting a good number of features would improve the performances of the model. However, the idea behind selecting only 40,000 top frequent features for each n-gram type in all language pairs and choosing the widely used ML classifiers is to keep the proposed method simple but robust.

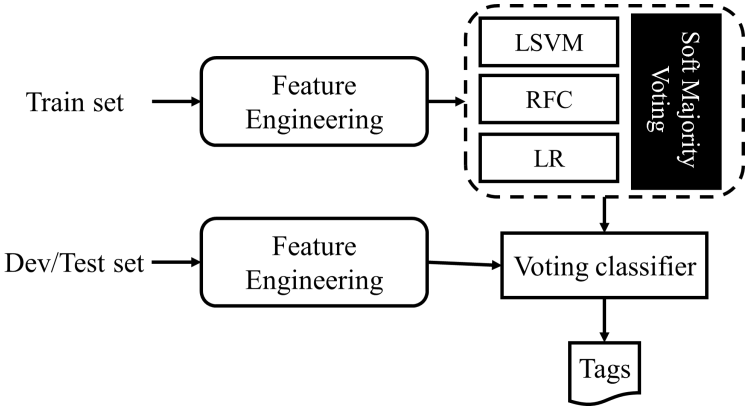


Figure 2: Voting classifier

Table 2
Statistics of Datasets

Tasks	Language	Labels	Train set	Dev set	Test set
Task 1	**Tamil**	OFF	1,153	—	118
		NOT	4,724		536
		Not-Tamil	3		0
Task 2	**Ta-En**	OFF	1,980	475	395
		NOT	2,019	465	605
	Ma-En	OFF	1,952	478	325
		NOT	2,047	473	675

4. Experiments and Results

Datasets provided by the HASOC-DravidianCodeMix shared task [21, 22] organizers are collected from social media. They contain Train, Development (Dev) and Test sets which includes Tamil text in Tamil script for Task 1 and code-mixed texts in Ta-En and Ma-En language pairs for Task 2. The statistics of the datasets given in Table 2 illustrate that the dataset is highly imbalanced which make it more challenging.

The performances of the models are evaluated and ranked by the organizers based on the average weighted F1-scores. Table 3 presents the performances of ML classifiers on Development set computed using Sklearn library for Task 2 only as Development set is not provided for Tamil language in native script for Task 1. The results illustrate that the individual classifiers and their ensemble obtained very competitive performances for both the language pairs. However, for the Development set, LR and RF classifiers outperformed the other classifiers for both the language pairs.

Results obtained on Test sets show that LR classifier outperformed other classifiers for Tamil and Ta-En language pair with average weighted F1-scores of 0.582 and 0.678 respectively. However, RF classifier obtained highest results for Ma-En language pair with average weighted F1-score of 0.762. The detailed performances of ML classifiers evaluated on Test sets are given in Table 4.

Table 3
Results of Task 2 for the Development set

Language	Classifier	Precision	Recall	F1-score
Ta-En	**LR**	**0.881**	**0.881**	**0.881**
	LSVM	0.865	0.865	0.865
	RF	0.864	0.860	0.859
	Ensemble	0.877	0.877	0.877
Ma-En	LR	0.756	0.755	0.755
	LSVM	0.751	0.748	0.747
	RF	**0.795**	**0.784**	**0.783**
	Ensemble	0.785	0.780	0.780

Table 4
Results of Task 1 and Task 2 for the Test set

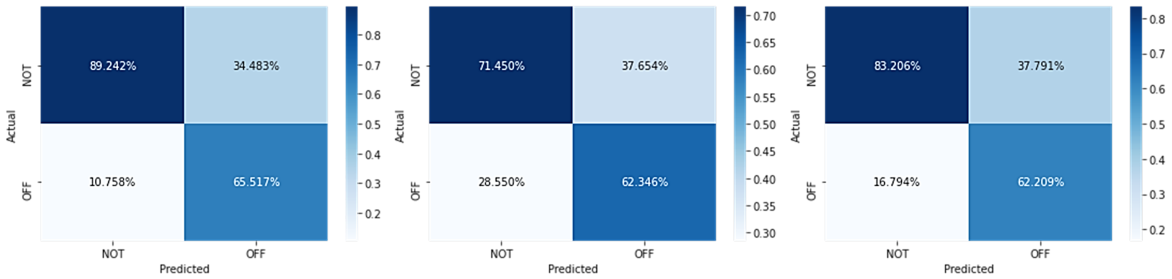
Task	Language	Classifier	Precision	Recall	F1-score	Rank
1	**Tamil**	**LR**	**0.850**	**0.861**	**0.852**	**2**
		LSVM	0.839	0.852	0.843	-
		RF	0.830	0.846	0.811	-
		Ensemble	0.847	0.861	0.842	-
2	**Ta-En**	**LR**	**0.679**	**0.685**	**0.678**	**1**
		LSVM	0.665	0.672	0.666	-
		RF	0.618	0.604	0.608	-
		Ensemble	0.677	0.681	0.678	-
	Ma-En	LR	0.759	0.737	0.743	-
		LSVM	0.754	0.724	0.731	-
		RF	**0.764**	**0.760**	**0.762**	**2**
		Ensemble	0.768	0.749	0.754	-

The confusion matrices for the best performing classifiers for each task are shown in Figure 3. LR classifiers have performed well for Task 1 and code-mixed Ta-En language pair of Task 2 and RF classifier has performed well for Ma-En language pair of Task 2 and the corresponding confusion matrices are shown in Figure 3a, 3b and 3c respectively. Analysis of the confusion matrices illustrate that classifiers were more successful in identifying NOT (not offensive) posts due to the higher number of samples in this category in the given datasets for both the tasks.

The comparison of the proposed strategy with the best performing teams of the shared task in terms of average weighted F1-score is presented in Figure 4. It can be observed that, on the average, team MUCIC outperformed all the other teams by obtaining average weighted F1-scores of 0.852, 0.678, and 0.762 and securing 2^{nd}, 1^{st}, and 2^{nd} ranks for Tamil (Task 1), Ta-En and Ma-En language pairs (Task 2) respectively.

5. Conclusion and Future Work

This paper describes the strategy proposed by the team MUCIC for the Dravidian-CodeMix-HASOC2021 shared task. In the proposed strategy, the top frequent char and word n-grams selected from the texts are stacked and converted to TFIDF vectors for training the ML classifiers.



3a. Tamil (Task1) 3b. Ta-En (Task2) 3c. Ma-En (Task2)

Figure 3: Confusion matrix of the best performing classifiers

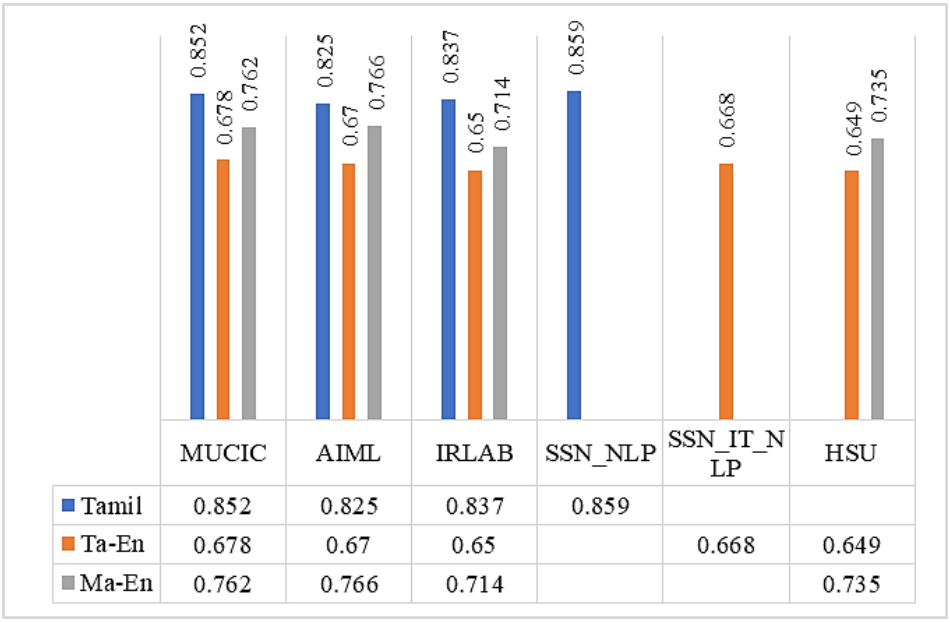


Figure 4: Comparison of F1-scores of the best performing teams

Team MUCIC participated in both Task 1 and Task 2 and secured 2nd, 1st, and 2nd ranks for Tamil and code-mixed Ta-En and Ma-En language pairs respectively. The proposed strategy outperformed most of the models submitted by the participants to the shared task and became one among the best performing teams. This work also proved the effectiveness of feature reduction (even a simple algorithm) algorithms for classification tasks. The statistical feature selection algorithms along with various feature sets to improve the performances of ML classifiers will be explored further.

Acknowledgments

Team MUCIC sincerely appreciate the organizers for their efforts to conduct this shared task.

References

[1] B. Fernandes, U. N. Biswas, R. T. Mansukhani, A. V. Casarín, C. A. Essau, The Impact of COVID-19 Lockdown on Internet use and Escapism in Adolescents, Revista de psicología clínica con niños y adolescentes 7 (2020) 59–65.

[2] S. Banerjee, A. Jayapal, S. Thavareesan, NUIG-Shubhanker@Dravidian-CodeMix-FIRE2020: Sentiment Analysis of Code-Mixed Dravidian Text using XLNet, in: FIRE, 2020.

[3] S. Banerjee, B. R. Chakravarthi, J. P. McCrae, Comparison of Pretrained Embeddings to Identify Hate Speech in Indian Code-Mixed Text, in: 2020 Second International Conference on Advances in Computing, Communication Control and Networking (ICACCCN), IEEE, 2020, pp. 21–25.

[4] C. PFAFF, Constraints on Language Mixing: Intrasentential Code-Switching and Borrowing in Spanish/English, Language. Journal of the Linguistic Society of America Baltimore, Md 55 (1979) 291–318.

[5] S. Suryawanshi, B. R. Chakravarthi, Findings of the Shared Task on Troll Meme Classification in Tamil, in: Proceedings of the First Workshop on Speech and Language Technologies for Dravidian Languages, Association for Computational Linguistics, Kyiv, 2021, pp. 126–132. URL: https://aclanthology.org/2021.dravidianlangtech-1.16.

[6] T. Mandl, S. Modha, A. Kumar M, B. R. Chakravarthi, Overview of the HASOC Track at FIRE 2020: Hate Speech and Offensive Language Identification in Tamil, Malayalam, Hindi, English and German, in: Forum for Information Retrieval Evaluation, 2020, pp. 29–32.

[7] B. R. Chakravarthi, R. Priyadharshini, N. Jose, T. Mandl, P. K. Kumaresan, R. Ponnusamy, R. Hariharan, J. P. McCrae, E. Sherly, et al., Findings of the Shared Task on Offensive Language Identification in Tamil, Malayalam, and Kannada, in: Proceedings of the First Workshop on Speech and Language Technologies for Dravidian Languages, 2021, pp. 133–145.

[8] B. R. Chakravarthi, R. Priyadharshini, P. Krishnamurthy, E. Sherly, et al., Proceedings of the First Workshop on Speech and Language Technologies for Dravidian Languages, in: Proceedings of the First Workshop on Speech and Language Technologies for Dravidian Languages, 2021.

[9] F. Balouchzahi, B. K. Aparna, H. L. Shashirekha, MUCS@DravidianLangTech-EACL2021: COOLI-Code-Mixing Offensive Language Identification, in: Proceedings of the First Workshop on Speech and Language Technologies for Dravidian Languages, 2021, pp. 323–329.

[10] B. R. Chakravarthi, P. K. Kumaresan, R. Sakuntharaj, A. K. Madasamy, S. Thavareesan, P. B, S. Chinnaudayar Navaneethakrishnan, J. P. McCrae, T. Mandl, Overview of the HASOC-DravidianCodeMix Shared Task on Offensive Language Detection in Tamil and

Malayalam, in: Working Notes of FIRE 2021 - Forum for Information Retrieval Evaluation, CEUR, 2021.

[11] 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 First Joint Workshop on Spoken Language Technologies for Under-resourced languages (SLTU) and Collaboration and Computing for Under-Resourced Languages (CCURL), 2020, pp. 202–210.

[12] 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, 2020, pp. 54–63.

[13] 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 First 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://aclanthology.org/2020.sltu-1.25.

[14] M. Zampieri, P. Nakov, S. Rosenthal, P. Atanasova, G. Karadzhov, H. Mubarak, L. Derczynski, Z. Pitenis, Ç. Çöltekin, SemEval-2020 Task 12: Multilingual Offensive Language Identification in Social Media (OffensEval 2020), in: Proceedings of the Fourteenth Workshop on Semantic Evaluation, 2020, pp. 1425–1447.

[15] D. Saha, N. Paharia, D. Chakraborty, P. Saha, A. Mukherjee, Hate-Alert@DravidianLangTech-EACL2021: Ensembling Strategies for Transformer-based Offensive Language Detection, in: Proceedings of the First Workshop on Speech and Language Technologies for Dravidian Languages, 2021, pp. 270–276.

[16] S. M. Jayanthi, A. Gupta, SJ_AJ@DravidianLangTech-EACL2021: Task-Adaptive Pre-Training of Multilingual BERT Models for Offensive Language Identification, in: Proceedings of the First Workshop on Speech and Language Technologies for Dravidian Languages, 2021, pp. 307–312.

[17] C. Vasantharajan, U. Thayasivam, Hypers@DravidianLangTech-EACL2021: Offensive Language Identification in Dravidian Code-Mixed YouTube Comments and Posts, in: Proceedings of the First Workshop on Speech and Language Technologies for Dravidian Languages, 2021, pp. 195–202.

[18] Z. Li, Codewithzichao@DravidianLangTech-EACL2021: Exploring Multilingual Transformers for Offensive Language Identification on Code Mixing Text, in: Proceedings of the First Workshop on Speech and Language Technologies for Dravidian Languages, 2021, pp. 164–168.

[19] B. Bharathi, SSNCSE_NLP@DravidianLangTech-EACL2021: Offensive Language Identification on Multilingual Code Mixing Text, in: Proceedings of the First Workshop on Speech and Language Technologies for Dravidian Languages, 2021, pp. 313–318.

[20] B. Dave, S. Bhat, P. Majumder, IRNLP_DAIICT@DravidianLangTech-EACL2021: Offensive Language Identification in Dravidian Languages using TF-IDF Char N-grams and MuRIL, in: Proceedings of the First Workshop on Speech and Language Technologies for Dravidian Languages, 2021, pp. 266–269.

[21] 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.

[22] 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.