Hate Speech and Offensive Language Identification on Multilingual code-mixed Text using BERT

Snehaan Bhawal[1], Pradeep Kumar Roy[2] and Abhinav Kumar[3]

[1]*Kalinga Institute of Industrial Technology, Odisha, India*
[2]*Indian Institute of Information Technology Surat, Gujarat, India*
[3]*Siksha 'O' Anusandhan, Deemed to be University, Bhubaneswar, Odisha, India*

Abstract

Hate Speech and Offensive Content detection in social media has been an active field of research for the last couple of years. For the majority of the world consisting of non-native English speakers, most of the time unofficial messages are written in code-mixed language in a combination of words in a native language with English text. The current study focuses on using Machine and Deep learning techniques for detection of Hate Speech and Offensive content in a Malayalam and Tamil code-mixed text collected from social media. The study showed that Deep learning models perform better than the machine learning models, specifically the implementation of BERT based transfer learning models performed best.

Keywords

Multilingual Text, Hate Speech, Deep Learning, Machine Learning, BERT

1. Introduction

Hate Speech is generally defined as content that expresses hate or prejudice against a particular group, ethnicity, religion, nationality or sexual orientation [1, 2, 3]. Social network platforms consists of a large amount of user-generated content, and due to being not moderated in nature, there is a widespread use of targeted hate speech against certain individuals, which has become a very critical issue [4, 5].

Humans can't always moderate the social media networks to read, identify, and deal with the hateful text that the platform generates in such high frequency affecting the users mentally. Thus, there is a need for automation, and it has already been established that detection of such content by automation has been successful to a certain extent. Davidson et al. [6] used Logistic Regression with n-grams TF-IDF features to perform classification of Offensive and Non-Offensive text. At the same time, in another paper, a neural network-based approach was presented by Badjatiya et al. [7], where they used GloVe embedding with CNNs and LSTMs to provide better results.

However, most of the research that has taken place over hate speech and offensive language detection is predominately for the English language [1]. In a country like India, with home

FIRE 2021,Forum for Information Retrieval Evaluation, December 13-17, 2021
mailtosnehaan@gmail.com (S. Bhawal); pradeep.roy@iiitsurat.ac.in (P. K. Roy); abhinavkumar@soa.ac.in (A. Kumar)
0000-0002-1072-5326 (S. Bhawal); 0000-0001-5513-2834 (P. K. Roy); 0000-0001-9367-7069 (A. Kumar)

to numerous regional languages, people have adapted to using a mix of regional and English languages to express themselves in social media [8, 9]. The current research is done upon bilingual texts, which contain words from both languages and are written in one script, called code-mixed text. While there is another way of combining the words in native writing with the English script, which is known as Script-mixed text. These are far more challenging to work with as it requires a different tokenization process compared to what we need for English texts. Examples of some popular code-mixed languages in India are Hinglish (Hindi and English), Tanglish (Tamil and English), Manglish (Malayalam and English), and a mixed language of Kannada and English [10].

Identifying Hate Speech in such code-mixed languages is much more challenging than in English [11] due to the absence of sufficient NLP resources. The models which are trained on a monolingual corpus might find it difficult to provide satisfactory results. This is because the system learns and recognizes the words provided in the given vocabulary while training. In the case of code-mixed text, many new words will be introduced which will not be present in the training vocabulary. The words are then marked as out of vocabulary token that makes no difference in the estimation of the model. Thus, the performance of the model decreases.

The current study focuses on Offensive language identification in code-mixed languages of Tanglish and Manglish with the data set provided in HASOC-Dravidian-CodeMix-FIRE2021 challenge. An overview of the dataset can be found here [12]. We have implemented a number of Machine learning and Deep learning models, including transfer learning models like BERT, to distinguish between the Offensive and Non-Offensive text.

The rest of the article is summarized as follows: Section 2 discusses the related works. Section 3, 3.1, 4 provides the task description, the pre-processing steps taken, followed by the explanation of the proposed methodology. The experimental results and discussion are explained in Section 5 and 6, respectively. Section 7 concluded the work by highlighting the limitations and future scope.

2. Literature Review

The use of Hate speech and Offensive language has become one of the major issues concerning the social networking platforms and hence received fruitful attention from many worldwide researchers [1, 2, 4, 8, 9]. Roy et al. [1] developed a deep learning-based framework to address the hate speech issue on Twitter. They used a Convolutional Neural Network to process the tweets and predict whether it were Hate or non-Hate. They considered only the tweets written in English language and hence unable to detect the tweets of multilingual texts, such as Tamil-English, Kannada-English and others. Badjatiya et al. [7] developed a deep learning model to classify the tweets into racist, sexist or neither category. Their model experimented on 16k labelled data and outperformed existing models. The main issues with the existing works are the coverage of the language. Most of the existing researches use an English dataset. However, currently, people prefer to post the message on the social platform in code-mixed languages like Hindi-English mixed, Tamil-English mixed and others.

Recent work by Kumar et al. [4] suggested a deep learning-based framework to classify the Tamil and Malayalam code-mixed YouTube comments into the offensive and non-offensive

categories. Many machines and deep learning models have experimented. The best result was obtained when a character n-gram tf-idf features passed to the dense neural network. Their model achieved the weighted F1-score value of 0.95. Suryawanshi et al. [13] developed the resources for Tamilmeme detection. The developed dataset consisted of two labels: troll and not_troll. A total of ten models were submitted, and the model with an F1-score value of 0.55 secured the first rank among them. Banerjee et al. [14] compared the performance of the pre-trained models on the Hinglish code-mixed dataset for predicting the Hate and non-Hate post.

3. Task and Data Description

The current study is an implementation and comparison of different Machine and Deep Learning models for a Hate Speech and Offensive Language detection system for Tamil and Malayalam code-mixed texts in English. The dataset consists of sentences collected from comments or posts from social media. Table 1 shows the overview of the data used in this analysis. There are two sets of data, Malayalam code-mixed and Tamil code-mixed data each consisting of code-mixed sentences with addition of various emojis in most of the cases.

3.1. Data Preprocessing

As the data was code-mixed with Malayalam or Tamil mixed with English, no stop-word removal was done. The text being informal in nature contained emojis and emoticons which were replaced with their respective textual meaning using data from the Unicode Consortium's emoji code repository by using the demoji library. This was then followed with the removal of punctuation, URLS, email-ids, hyperlinks and numeric data from the text.

4. Methodology

This section discusses the working of the implemented models in detail, the codes of which can be found in the GitHub repository[1]. In our current study, three different approaches were used as shown in Figure 1:

i Conventional Machine learning based models.
ii Neural Network based models
iii Transfer learning based models

4.1. Traditional ML Models

In traditional ML-based models, we looked into using a 1 - 5 gram word TF-IDF feature set. The extracted features were then fed to classifiers like Logistic Regression (LR), Naive Bayes (NB), Random Forest (RF), XGBoost (XGB), and Support Vector Machine (SVM). The performance of these models were evaluated in terms of precision, recall, and F1-score [15]. The detailed performance report of these models are provided in the Section 5.

[1]https://github.com/Sbhawal/HASOC-FIRE-2021-CODES

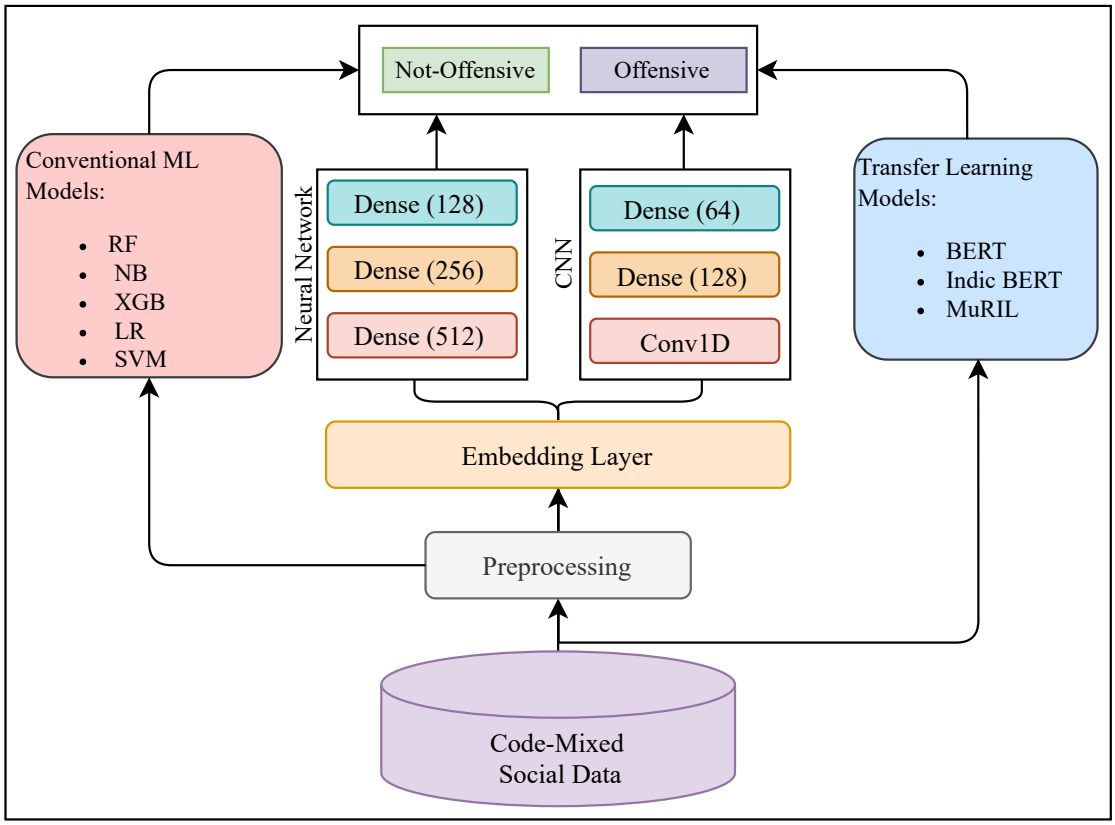

Figure 1: Framework used to predict the offensive post

Table 1
Distribution of Data in the Training and Validation classes

Data Set	Class	Offensive	Not Offensive	Total
Malayalam	Train	1952	2047	3999
Code-Mixed	Validation	478	473	951
Tamil	Train	2019	1980	3999
Code-Mixed	Validation	465	465	940

4.2. Neural Network based models

In neural network-based models, the 1 to 5 grams TF-IDF features extracted while working with the Machine Learning models were used again as an input to a simple deep neural network (DNN) model. This model consisted of four fully connected layers in sequential order, with 512, 256, 128 and 1 neurons in the first, second, third and fourth (output) layers. Due to classification between two distinct labels, only one output neuron was used to identify the outputs. The hidden neurons were set up with the ReLU activation function. In contrast, the output neuron was set up with sigmoid activation function with Adam and binary-cross-entropy as the respective

chosen optimizer and loss function.
The second experimented neural model is Convolutional Neural Network (CNN) [16]. The CNN consisted of one Conv1D layer followed by a Global Max Pooling and a Dropout layer connected to a fully connected sequential network with two hidden layers of 128 and 64 neurons, respectively, The activation function for the hidden neurons were chosen to be ReLU. The output was a single neuron with sigmoid activation. An embedding layer was used as the input layer with the embedding dimension set to 50 and the input length set to 120. Therefore, a (120, 50) dimensional embedding matrix was given as an input to CNN. The Convolutional layer consisted of 64 filters with a kernel size of three.
Our final neural network based model was a Bidirectional Long Short-Term Memory model (Bi-LSTM), consisting of 256 memory units followed by Global Max Pooling and Batch Normalization. The input layer was an embedding layer with 50 dimensions and length padded to 120 like the previous model. Two fully connected dense layers served as the hidden layers comprising of 20 and 10 neurons, respectively with ReLU as the activation function, which was then connected to a single neuron as the output layer with sigmoid activation.
Subsequent Hyper-parameter tuning was done for the described models to check for the optimal performance by adjusting the optimizer, learning rate and embedding dimensions. Our experiments led to the best result with a learning rate set to 0.0001 with the optimizer set as Adam. The embedding dimension was set to 50 as it gave the best result. Due to binary classification and the overall balanced nature of the data set, the loss function was kept to be binary cross-entropy and sigmoid activation function for the output neuron.

4.3. Transfer Learning

This study has implemented BERT (Bidirectional Encoder Representations from Transformers) models to work with these models' transfer learning capabilities. For these models, no preprocessing was done. Three different variants of BERT models were studied.

i BERT (multilingual) [17].
ii IndicBERT [18].
iii Multilingual Representations for Indian Languages (MuRIL) [19].

The BERT [17] multilingual model was trained on 102 languages with masked language modelling. The case-sensitive model was chosen, as no prior data pre-processing was done in case of transfer learning models. IndicBERT [18] is a multilingual ALBERT model, pretrained exclusively on a corpus of 12 major Indian languages. Compared to other such BERT based models, IndicBERT is comparatively smaller and has much less number of parameters. We used ktrain [20] libraries to develop the IndicBERT model. The last model that we used is MuRIL(Multilingual Representations for Indian Languages) [19]. MuRIL is a BERT model trained over a monolingual corpus of 17 Indian languages along with their translated and transliterated counterpart. The differentiating factor between this and the previous model is that IndicBERT is trained only on the native Indian scripts. In contrast, MuRIL is trained on traditional scripts as well their transliterated corpus in roman script. The benefit of this will be evident in our experiment, which deals with code-mixed data of Indian and English language written strictly in roman script.

Table 2
Results of Traditional ML Models on Malayalam and Tamil code-mixed validation data set

Model	Class	Malayalam Code-Mixed			Tamil Code-Mixed		
		Precision	Recall	F1-score	Precision	Recall	F1-score
LR	Offensive	0.73	0.65	0.69	0.81	0.86	0.83
	Not Offensive	0.68	0.75	0.72	0.85	0.79	0.82
	Weighted Avg	0.70	**0.70**	**0.70**	**0.83**	**0.82**	**0.82**
RF	Offensive	0.78	0.51	0.61	0.78	0.87	0.82
	Not Offensive	0.63	0.85	0.73	0.85	0.76	0.80
	Weighted Avg	**0.71**	0.68	0.67	0.81	0.81	0.81
NB	Offensive	0.71	0.66	0.69	0.84	0.77	0.81
	Not Offensive	0.68	0.73	0.71	0.78	0.85	0.82
	Weighted Avg	0.70	**0.70**	**0.70**	0.81	0.81	0.81
XGB	Offensive	0.81	0.33	0.47	0.67	0.93	0.78
	Not Offensive	0.58	0.92	0.71	0.89	0.52	0.66
	Weighted Avg	0.70	0.63	0.59	0.77	0.73	0.72
SVM	Offensive	0.72	0.60	0.66	0.76	0.88	0.82
	Not Offensive	0.66	0.77	0.71	0.85	0.72	0.78
	Weighted Avg	0.69	0.68	0.68	0.81	0.80	0.80

5. Results

This section presents the results of all our experiments done during this study, as mentioned in Section 2. The results shown below corresponds to the model prediction on the validation data and are shown in terms of precision, recall, and F1-score belonging to OFF (Offensive) or NOT (Not Offensive) class. A model is said to be the best if it reports the highest weighted average in terms of precision, recall, and F1-score. The best results for the particular data set are presented in bold for each different model used in this study.

Traditional ML models were built using 1 to 5-gram character TF-IDF features which included the following models, LR, RF, NB, XGB and SVM. Their results are shown in Table 2 respectively. In the Malayalam code-mixed data set, the LR classifier gave a better performance with recall and F1 of 0.70. Similarly, in Tamil code-mixed text, the LR classifier performed the best and reported precision of 0.83 with recall and an F1-score of 0.82.

Results of the neural network models are presented in Table 3. It is seen that a simple DNN provided the best results in the case of the Malayalam Code Mix data with a precision of 0.75 with recall and F1-Score being 0.74. In Tamil Code Mix data, CNN showed the best performance with precision reaching 0.90 with Recall and F1-Score of 0.89.

In Table 4 the results of using different BERT models are presented. In both Malayalam and Tamil Data, it is seen that the MuRIL model performed the best among the other models. In Malayalam data, the precision was 0.79 with recall and F1-Score being 0.78, and for Tamil data, precision, recall and F1-Score were 0.91, which was the highest among all experimented models

Table 3
Results of Neural Network based models on Malayalam and Tamil code-mixed data set

Model	Class	Malayalam Code-Mixed			Tamil Code-Mixed		
		Precision	Recall	F1-score	Precision	Recall	F1-score
DNN	Offensive	0.73	0.67	0.70	0.84	0.83	0.83
	Not Offensive	0.69	0.75	0.72	0.83	0.83	0.83
	Weighted Avg	0.71	0.71	0.71	0.83	0.83	0.83
DNN+ Emb	Offensive	0.79	0.65	0.72	0.84	0.92	0.88
	Not Offensive	0.70	0.82	0.76	0.91	0.82	0.86
	Weighted Avg	**0.75**	**0.74**	**0.74**	0.87	0.87	0.87
CNN	Offensive	0.78	0.61	0.68	0.95	0.84	0.89
	Not Offensive	0.68	0.83	0.74	0.85	0.95	0.90
	Weighted Avg	0.73	0.72	0.71	**0.90**	**0.89**	**0.89**
Bi-LSTM	Offensive	0.77	0.52	0.62	0.92	0.76	0.83
	Not Offensive	0.64	0.84	0.72	0.79	0.93	0.86
	Weighted Avg	0.70	0.68	0.67	0.86	0.84	0.84

Table 4
Results of Transfer Learning based models on Malayalam and Tamil code-mixed validation data set

Model	Class	Malayalam Code-Mixed			Tamil Code-Mixed		
		Precision	Recall	F1-score	Precision	Recall	F1-score
BERT	Offensive	0.75	0.73	0.74	0.88	0.92	0.90
	Not Offensive	0.74	0.75	0.74	0.92	0.87	0.89
	Weighted Avg	0.74	0.74	0.74	0.90	0.90	0.90
Indic BERT	Offensive	0.71	0.69	0.70	0.78	0.82	0.80
	Not Offensive	0.70	0.72	0.71	0.81	0.76	0.78
	Weighted Avg	0.71	0.71	0.71	0.79	0.79	0.79
MuRIL	Offensive	0.82	0.72	0.77	0.92	0.91	0.92
	Not Offensive	0.75	0.84	0.79	0.91	0.92	0.91
	Weighted Avg	**0.79**	**0.78**	**0.78**	**0.91**	**0.91**	**0.91**

6. Discussion

Among all experimented models, the MuRIL-a transfer learning model, performed the best for both Malayalam and Tamil code-mixed data. The experimental outcomes show that the traditional machine learning models are unable to understand the context of the message and hence may not be a good choice for this task. A simple Deep Neural Network (DNN) with an embedding layer performed better than most of the machine learning models (Tables 2, 3). Although some of their performance came near those of neural network models, we were dealing mostly with text data consisting of single sentences. For multiple sentence texts, a neural network with the ability to hold some memory like LSTM would have outclassed the machine learning models [21].

As shown in Table 4, the IndicBERT model is not able to perform as good as the multilingual BERT model. This may happen because the data set consisted of code-mixed data in the Roman script only. If there were any text written in the traditional script, then the multilingual BERT

Table 5
Test Data Prediction Results on selected models

Model	Class	Malayalam Code-Mixed			Tamil Code-Mixed		
		Precision	Recall	F1-score	Precision	Recall	F1-score
LR	Offensive	0.51	0.65	0.57	0.52	0.56	0.54
	Not Offensive	0.81	0.70	0.75	0.70	0.66	0.68
	Weighted Avg	0.71	0.69	0.69	0.62	0.62	0.62
DNN+ Emb	Offensive	0.49	0.65	0.56	0.55	0.53	0.54
	Not Offensive	0.80	0.68	0.73	0.70	0.72	0.71
	Weighted Avg	0.70	0.67	0.67	0.64	0.64	0.64
CNN	Offensive	0.55	0.52	0.53	0.56	0.50	0.53
	Not Offensive	0.78	0.79	0.78	0.69	0.75	0.72
	Weighted Avg	0.70	0.70	0.70	0.64	0.65	0.64
BERT	Offensive	0.56	0.65	0.60	0.61	0.84	0.71
	Not Offensive	0.82	0.75	0.78	0.73	0.44	0.55
	Weighted Avg	0.73	0.72	0.72	0.67	0.64	0.63
MuRIL	Offensive	0.55	0.60	0.58	0.58	0.43	0.50
	Not Offensive	0.80	0.77	0.78	0.68	0.80	0.74
	Weighted Avg	0.75	0.72	0.73	0.68	0.67	0.64

model would have treated most of the tokens as an unknown token which would had affected the model performance—benefiting the IndicBERT model as it was trained on monolingual Indian scripts. Finally, MuRIL, which was trained on a corpus of both traditional script and transliterated one, performed better than all the models.

The above reported results (Tables 2, 3, 4) were based on the predictions done over the validation data set. While using the test data, the proposed MuRIL model achieved the precision, recall and F1-score value of 0.679, 0.673 and 0.636, respectively for Tamil code-mixed data, while on the Malayalam code-mixed data, the precision, recall and F1-score value is 0.752, 0.727, and 0.734, respectively for the best case.

The models were re-experimented with labelled test data, and the obtained results with different machine learning, neural network and transfer learning models are shown in Table 5. Similar to the results on the validation data, MuRIL -a transfer learning model produces the best prediction outcomes in terms of weighted average precision, recall and F1-score for both Tanglish and Manglish test dataset.

7. Conclusion

Hate speech and offensive language detection is still a challenge for low resource and code-mixed languages in NLP. We implemented various machine learning, deep learning and transfer learning models to find the best suitable model for code-mixed Tamil and Malayalam datasets. The results reported by the models show the deep learning models. Specifically, the pre-trained models outperformed the machine learning models. The MuRIL model performed the best reporting weighted F1-score of 0.636 in Tamil code-mixed data. The same model provided a weighted F1-score of 0.734 in Malayalam code-mixed data. On test data, the BERT and MuRIL

both transfer learning model yielded almost similar outcomes. In the future, a better model can be built by some additional preprocessing steps on the dataset to achieve better prediction accuracy.

References

[1] P. K. Roy, A. K. Tripathy, T. K. Das, X.-Z. Gao, A framework for hate speech detection using deep convolutional neural network, IEEE Access 8 (2020) 204951–204962.

[2] S. Saumya, A. Kumar, J. P. Singh, Offensive language identification in dravidian code mixed social media text, in: Proceedings of the First Workshop on Speech and Language Technologies for Dravidian Languages, 2021, pp. 36–45.

[3] 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.

[4] A. Kumar, S. Saumya, J. P. Singh, Nitp-ai-nlp@ hasoc-dravidian-codemix-fire2020: A machine learning approach to identify offensive languages from dravidian code-mixed text., in: FIRE (Working Notes), 2020, pp. 384–390.

[5] A. Kumar, S. Saumya, J. P. Singh, Nitp-ai-nlp@ hasoc-fire2020: Fine tuned bert for the hate speech and offensive content identification from social media., in: FIRE (Working Notes), 2020, pp. 266–273.

[6] T. Davidson, D. Warmsley, M. Macy, I. Weber, Automated hate speech detection and the problem of offensive language, in: Proceedings of the International AAAI Conference on Web and Social Media, volume 11, 2017.

[7] P. Badjatiya, S. Gupta, M. Gupta, V. Varma, Deep learning for hate speech detection in tweets, in: Proceedings of the 26th international conference on World Wide Web companion, 2017, pp. 759–760.

[8] S. M. Jayanthi, A. Gupta, Sj_aj@ dravidianlangtech-eacl2021: Task-adaptive pre-training of multilingual bert models for offensive language identification, arXiv preprint arXiv:2102.01051 (2021).

[9] 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.

[10] B. R. Chakravarthi, R. Priyadharshini, N. Jose, A. Kumar M, T. Mandl, P. K. Kumaresan, R. Ponnusamy, H. R L, J. P. McCrae, E. Sherly, 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, Association for Computational Linguistics, Kyiv, 2021, pp. 133–145. URL: https://aclanthology.org/2021.dravidianlangtech-1.17.

[11] 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.

[12] 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.

[13] 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.

[14] S. Banerjee, B. Raja Chakravarthi, J. P. McCrae, Comparison of pretrained embeddings to identify hate speech in indian code-mixed text, in: 2020 2nd International Conference on Advances in Computing, Communication Control and Networking (ICACCCN), 2020, pp. 21–25. doi:10.1109/ICACCCN51052.2020.9362731.

[15] D. Tripathi, D. R. Edla, R. Cheruku, V. Kuppili, A novel hybrid credit scoring model based on ensemble feature selection and multilayer ensemble classification, Computational Intelligence 35 (2019) 371–394.

[16] P. K. Roy, Multilayer convolutional neural network to filter low quality content from quora, Neural Processing Letters 52 (2020) 805–821.

[17] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, arXiv preprint arXiv:1810.04805 (2018).

[18] D. Kakwani, A. Kunchukuttan, S. Golla, G. N.C., A. Bhattacharyya, M. M. Khapra, P. Kumar, IndicNLPSuite: Monolingual Corpora, Evaluation Benchmarks and Pre-trained Multilingual Language Models for Indian Languages, in: Findings of EMNLP, 2020.

[19] S. Khanuja, D. Bansal, S. Mehtani, S. Khosla, A. Dey, B. Gopalan, D. K. Margam, P. Aggarwal, R. T. Nagipogu, S. Dave, S. Gupta, S. C. B. Gali, V. Subramanian, P. Talukdar, Muril: Multilingual representations for indian languages, 2021. arXiv:2103.10730.

[20] A. S. Maiya, ktrain: A low-code library for augmented machine learning, arXiv preprint arXiv:2004.10703 (2020). arXiv:2004.10703.

[21] P. K. Roy, J. P. Singh, S. Banerjee, Deep learning to filter sms spam, Future Generation Computer Systems 102 (2020) 524–533.