Detection Offensive Tamil Texts using Machine Learning and Multilingual Transformers Models

Malliga Subramanian[1], Kogilavani Shanmuga Vadivel[1], Antonette Shibani[2], Adhithiya G J[1], Deepti R[1], Gowtham Krishnan S[1]

[1]*Department of Computer Science and Engineering, Kongu Engineering College, Erode, Tamil Nadu, India*
[2] *University of Technology Sydney, Ultimo, Australia*

Abstract

Social media has facilitated an exponential increase in the distribution of hostile and toxic content in today's world. To tackle this issue, automated detection of such offensive content including hate speech, inflammatory, and abusive language have recently piqued the interest of many in the Natural Language Processing community. In this paper, we present machine learning and multilingual transformer models to automatically classify Tamil language comments as Offensive or Not-Offensive texts. The models make use of the dataset supplied for the Hate Speech and Offensive Content Identification (HASOC) challenge in Dravidian Languages under FIRE 2021. Among the proposed offensive language identification models, RoBERTa outperforms the others with an accuracy of 91.39%.

Keywords
Hate Speech, Offensive, Toxicity, Social Media, Youtube, Tamil, Machine Learning, RoBERTa, HASOC

1. Introduction

People around the world increasingly use social media platforms to share information and communicate with each other as part of their daily interactions. The widespread popularity of social media and micro-blogging platforms enhances connectivity among people, but it can also negatively affect their lives. Hateful and offensive comments have proliferated in social media, often by toxic users who hide under anonymity features of these platforms bypassing editorial control [1, 2]. If not curbed early, such toxic behavior can have a ripple effect and dissuade other people from being a part of the community [3] The hostile environment can stop them from expressing themselves freely due to the fear of being abused or harassed. Offensive language, hate speech and other objectionable content on the internet hence pose a threat to the well-being of our society [4].

The adverse societal impact of the spread of offensive language in social media is well recognized. However, the sheer scale of the issue and lack of regulation makes it a hard problem to tackle. Many countries outlaw hate speech on social media, with the caveat that it does not target any specific group or incite criminal behavior. Because hate speech shapes public opinion, platforms including YouTube, Facebook, and Twitter, have policies and tools in place to moderate hate speech content and related offensive behaviour to curb its ill effects on society [5]. One common approach is to employ automated methods that identify offensive language in order to remove/ hide them from other users [6].

Systems for the automatic identification of hate and offensive language in Natural Language Processing (NLP) usually fall under one of the following categories: i) feature-based linear classifiers [7, 8], ii) neural network architectures such as CNN, RNN [9-11] and, iii) fine-tuned pre-trained language models, such as BERT (Bidirectional Encoder Representation from Transformer) and RoBERTa (Robustly Optimized

Forum for Information Retrieval Evaluation, December 13-17, 2021, India
EMAIL: mallinishanth72@gmail.com(Malliga Subramanian);kogilavani.sv@gmail.com(Kogilavani Shanmuga Vadivel);
antonette.shibani@gmail.com (Antonette Shibani)
ORCID: https://orcid.org/0000-0003-3263-0376 (Malliga Subramanian)

BERT Pretraining Approach), among others [12, 13]. Building on this work, we present machine learning and multilingual transformer models to automatically detect YouTube user comments as offensive or Not-Offensive texts for one of the tasks assigned under HASOC-Offensive Language Identification in Dravidian Code-Mix FIRE 2021[1]. We specifically focus on Tamil, a classical language originating in India and spoken in Tamil Nadu, India's southernmost state, as well as Sri Lanka, Malaysia, and Singapore

The remainder of the paper is structured as follows: Section 2 reviews background work on hate speech and offensive language detection by surveying existing research on the topic. Section 3 provides a task description, a summary of the dataset used in the study and 4 explains our proposed models, followed by results and discussion in Section 4. Finally, we conclude the paper in Section 5 by summarizing the findings and implications of the work.

2. Literature Survey

The sheer volume and the growing number of people who use social media make it implausible for manual detection and removal of objectionable content through moderators. There is a high demand for tools and techniques that automatically detect offensive content on the internet to reduce its spread quickly and effectively. Recent years have witnessed a surge in the development of automated systems that filter offensive language on social media platforms, with identified challenges in classifying nuances due to the subjective and context-dependent nature of texts [6, 14]. Additional challenges emerge when working with texts containing more than one language –referred to as code-mixed data henceforth, which arises from multilingual users. Many previous systems eliminate data in languages other than English [14, 15], inducing scarcity in research in this area for low-resource languages.

As the user-generated content in social-media is typically code-mixed and not well studied for under-resourced languages, recent works have involved the development of systems on offensive text detection for such languages [16-18]. Additionally, the findings of the first shared task on Offensive Language Identification in Tamil, Malayalam, and Kannada, which relied on a thoroughly annotated data set based on human judgments was presented by [16]. The task setup allowed for testing of the model in multilingual contexts as well as the code-mixing phenomenon.

A novel attempt in [17] included a multimodal classification problem in the shared task "Troll Meme Classification in Tamil" introducing an enhanced TamilMeme dataset that now includes Tamil text from memes. This work while presenting a multimodal classification problem also discussed difficulties in natural language processing of a low-resourced language and code-mixed dataset. Moreover, work by [18] describe the shared task of machine translation for Dravidian (Tamil) languages presented at the first workshop on Speech and Language Technologies for Dravidian Technologies, whereby the best performing systems showcased a high Bilingual Evaluation Understudy Score despite a shortage of training data. Further work is essential in the detection of offensive language in under-resourced Tamil, Malayalam and Kannada languages to add to this growing area.

Next, we highlight the approaches based on machine learning and natural language processing that have made significant progress in automatically detecting offensive language on web platforms. Traditional machine learning approaches employ features such as word-level and character-level n-grams, amongst others. Using a multi-class classifier, Davidson et. al.[19] classified tweets as hate speech, offensive, neither hate nor offensive using Naive Bayes, Decision Trees, Random Forests etc. with 5-fold cross validation and claimed model performance metrics as follows: precision 0.91, recall 0.90, and F1 score 0.90. Gibert et al. [20] created a manually labelled hate speech dataset from Stormfront, a white supremacist online forum that contains hateful and non-hateful sentences. To annotate the test data based on hand-annotated training data, Support Vector Machines (SVM), Convolutional Neural Networks (CNN), and Recurrent Neural Networks have been used.

When comparing linear classifiers to neural networks, the results were inconsistent across datasets and architectures, with linear classifiers proving to be very competitive, if not superior. Systems based on pre-trained language models, on the other hand, have proven to have the best performance in this area, achieving new state-of-the-art results. However, one limitation in the case of these pre-trained models is that they are suited only for general-purpose language comprehension tasks because of their training language variety. To test machine learning models for hate speech classification, the authors have provided hate speech benchmark datasets [21]. In addition, the advantages and disadvantages of single and hybrid

[1]http://fire.irsi.res.in/fire/2021/home

machine learning methods in the classification of hate speech were discussed in [21]. Besides the existing approaches and datasets for hate speech detection, MacAvaney et al. [22] investigated the challenges of automatic approaches for hate speech detection online. This study proposed a multi-view SVM approach that outperforms neural methods while being simpler and more interpretable. The experiments use datasets such as HatebaseTwitter, Stormfront, and TRAC [22]. Using bidirectional LSTM, Garain and Basu [23] described their system to detect offensive language in Twitter.

Although numerous feature extraction methods have been utilised in machine learning-based approaches, what they require in common is a well-defined feature extraction strategy. To increase the performance of hate speech and offensive content detection models, neural network models now use text representation and deep learning methodologies such as CNNs , Bi-directional Long Short-Term Memory Networks (LSTMs), and BERT [24]. A study in [24] employed pre-trained BERT and multilingualBERT to detect hate speech and offensive content in English, German, and Hindi. The authors of [12] experimented with the following classifiers: a linear model containing features from word unigrams, word2vec, and Hatebase, word-based LSTM and a fine-tuned BERT. For this work, the Offensive Language Identification Dataset (OLID) is gathered via the Twitter API by searching a specific set of terms. Another study proposed BERT, a transfer learning approach based on an existing pre-trained language bidirectional encoder representation models [25]. In particular, this work studied BERT's ability to capture hateful contexts in social media content using new fine-tuning strategies based on transfer learning, and employed two publicly available datasets annotated for racism, sexism, hate, or offensive content on Twitter to assess the proposed approach.

While multilingual transformer models were seen to be successful for the shared task provided in [16], a few machine learning models also performed well using hybrid approaches for feature selection/extraction from texts rather than single feature selection/extraction algorithms. In comparison to utilising a transformer, the advantage of employing machine learning models with feature selection techniques is that the model remains simple for easier interpretation, and hence form the basis of the work we describe in this paper. We demonstrate four different machine learning algorithms to categorize YouTube comments as offensive or Not-Offensive in addition to RoBERTa. Based on the no free lunch theorem [26] and our review of the current works with disparity in datasets, we note that, a method that works well for one dataset might not be appropriate for another. This means that there might not be a single classifier which best performs on all kinds of datasets and hence, we apply several different classifiers on a feature vector to observe which one provides better results. We provide more details of our approach in the next section.

3. Materials and Methods

3.1 Taskset Description

Offensive texts in this study adopt the common definition for offensive language often expressed as ‘flames’, which refers to “offensive messages or remarks that in some circumstances are inappropriate, exhibit a lack of respect towards certain groups of people or are just rude in general”. The dataset used to identify offensive language content consists of code-mixed data of Tamil comments collected from YouTube media, made available through HASOC-Offensive Language Identification track in Dravidian Code-Mix FIRE 2021[1]. The comments are in code-mixed form (Mixture of Native and Roman Script) comprising of Tamil, English and Malayalam languages [27] .

There are comments in Tamil, English, and Malayalam in the dataset, and none of them are totally in English or Malayalam, but are mixed with Tamil. The comments in dataset contains more than one sentence, but the corpora's average sentence length is one [28]. Each comment in the dataset was annotated as either Offensive (OFF), Not-Offensive (NOT) or Non-Tamil. The dataset contained 5880 Tamil texts for training and 654 Tamil texts for testing with class labels as Offensive, Not-Offensive and Not-Tamil. The training set has 1153 texts under Offensive and 4724 as Not-Offensive texts. Three more texts mixed with Tamil and English/Malayalam were omitted. Since the dataset was imbalanced, we augmented available data by shuffling the texts to generate new texts. This was done by shuffling the words in each sentence to generate a new one [29]. We also reduced the imbalance by applying Synthetic Minority

Oversampling TEchnique (SMOTE) [30], which is an oversampling approach that creates synthetic samples from the minority class. SMOTE is used to generate a synthetically or nearly class-balanced training set, which is subsequently utilized to train the classifier and works by selecting instances in the feature space that are near together. Sample Offensive and Not-Offensive texts from the dataset are presented in Table 1.

Table 1
Sample training texts from the dataset

Documents	Texts	Label
Document[10]	படம் அழகாக இருக்குங்க அத விட எதார்த்தமாயிருக்குங்க	NOT
Document[11]	இந்த படம் மாபெரும் வெற்றி பெற்று தமிழ் சினிமாவில் ஆதிக்கம் செலுத்தும் ,,,இது போன்ற மேலும் பல படங்களை இயக்க வாழ்த்துக்கள்	NOT
Document[13]	தமிழக மக்கள் சார்பாக வாழ்த்துக்கள்... சாதிகள் இல்லையடி பாப்பா. சலுகைகள் நிறைய வேண்டுமடி ((ஓட்டுக்காக)) பாப்பா..	NOT
Document[352]	இப்புடி படிச்ச பசங்க எல்லாம் வேனானு ஓடி ஓடி தான்யா நாடே நாசமா போயிருச்சு....	OFF
Document[62]	குரூமா குருப்பு குண்டி வெந்தே சாகபோரானுக வாழ்த்துக்கள் மோகன் சார்....	OFF

3.2 Pre-processing and Feature Extraction

Preprocessing the corpus (or data cleaning) is the first step to build any classifier. As the corpus contained emojis, punctuation characters and texts that are not in Tamil, these were removed by preprocessing them. The emojis were converted into text using the Emoji for Python library and CountVectorizer was used to convert the text corpus to a vector of term/token counts. The CountVectorizer from Python scikit-learn library looks after controlling n-gram size, custom preprocessing, tokenization, stop words, and vocabulary size, making it a versatile feature representation module for texts. The result of the vectorizer is an encoded vector with the vocabulary length and an integer count of each word's appearances in the document. Sample results from CountVectorizer for the dataset take is shown in Table 2. Here, we took the top 1050 features with word length more than or equal to 3. Note that CountVectorizer does not store these words as strings. Rather, they are assigned an index value. In this case, ‘படம் ' has index 0, ‘அழகாக' has index 1, and so on. Using fit() and transform() methods, each vector's value is calculated for each comment in the training set before inputting them to the classifiers.

Table 2
Results from Count Vectorizer

Text	படம்	அழகாக	இருக்குங்க	எதார்த்தமாயிருக்குங்க	பாப்பா
Document[10]	1	1	1	1	0
Document[11]	1	0	0	0	0
Document[13]	0	0	0	0	2

3.3. Proposed Classifiers

Having described the text to feature transformation, we present the classification algorithms used for offensive text detection in our study. We propose five classifiers based on machine learning namely, Naïve Bayes Multinomial classifier, SVM, Logistic Regression and KNN and a transformer model called

RoBERTa. The output from count vectorizer is fed as input to these classifiers and the output is the classification identified for each text. The scikit-learn libraries in Python have been used to implement the proposed models.

We now explain the rationale behind the use of the above models. The Naive Bayes Multinomial Classifier is a probabilistic model and a specialized version of the Naive Bayes algorithm. Simple Naive Bayes models a document for the presence or absence of specific words, whereas Multinomial classifier explicitly models the word counts and works well on small amounts of training data and trains relatively quickly when compared to other models [31]. Support-vector machines (SVM) are supervised classification algorithms that learn from training data to construct an optimal hyperplane that separates the categories while classifying new data. It handles a large number of samples well as SVMs are designed to find a hyperplane that maximizes the marginal distance between classes. In binary classification, the support vectors generate a hyperplane that divides the cases into two non-overlapping classes. SVM classifiers perform admirably in text classification tasks [31]. SVC() from the scikit-learn library has been used in our experiments with Radial Basis Function (RBF) as the kernel function.

Another model called Logistic regression uses a sigmoid function to explain the relationship between one independent variable and one or more independent variables. Here, we employed L2 regularization for this model to deal with data that contains a binary dependent variable, such as offensive or not offensive It calculates the probability of an output by combining the independent or prediction variables in a linear fashion. KNN is a simple text classification algorithm that categorizes new data by comparing it to all available data using some similarity measure. While KNN is a simple algorithm that applies to both classification and regression tasks and makes no assumptions about the data, it is memory and time intensive because all training data must be stored. All the classifiers give different values for performance metrics, as they perform differently based on their features discussed above.

RoBERTa is one of the most intriguing architectures derived from the BERT revolution. According to the authors of [32], while BERT exhibited a substantial performance improvement across various tasks, it was undertrained. RoBERTa improves the training technique by removing the next sentence prediction task from BERT's pre-training and introducing dynamic masking, which causes the masked token to change during the training epochs. Larger batch-training sizes have also been found to be more useful in the training operation. This enables RoBERTa to outperform BERT on the masked language modelling objective, resulting in higher downstream task performance.

3.4 Model implementation

Python programming language and its Sci-kit learn library have been used to implement the classification models. Google Co-laboratory (Colab) was used to train and test the proposed models. Colab is a fully cloud-based Jupyter notebook environment that doesn't require any desktop setup, hence it can run on any browser. The code-base is released open-source on Github.

4. Results and Discussion

In this section, we discuss the results of the classification models built for automatic detection of Offensive and Not-Offensive Tamil texts using the Dravidian Code-Mix FIRE 2021[1] dataset. We train each classifier using the features extracted from the training set and test the models using the test dataset provided.

4.1 Performance Metrics

The performance of the different models used for the classification problem have been evaluated using the following metrics: Accuracy, Precision, Recall and F1-Score [21]. These metrics commonly used for the evaluation of classifiers are defined as follows.

Accuracy is defined as the number of texts correctly classified as belonging to a specific class divided by the total number of texts in that class and is calculated by Equation (1).

$$\text{Accuracy} = (\text{TP} + \text{TN})/(\text{TP} + \text{TN} + \text{FP} + \text{FN}) \quad (1)$$

Where, TP = True positive (the number of correctly classified texts for each class), TN = True Negative (the number of texts correctly classified in other class except the correct class), FP = False Positive (number of texts misclassified in other class except the right class) and FN = False Negative (the number of texts misclassified in the relevant class) in the confusion matrix.

The number of texts correctly categorized as a certain class out of the total number of actual texts in that class is defined as Recall (also known as Sensitivity or True Positive Rate) and is computed using Equation (2).

$$\text{Recall} = \text{TP}/(\text{TP} + \text{FN}) \quad (2)$$

Precision (Positive Predictive Value) is defined as the number of texts accurately categorized as a specific class out of the total number of texts categorized as that class, and is given by Equation (3).

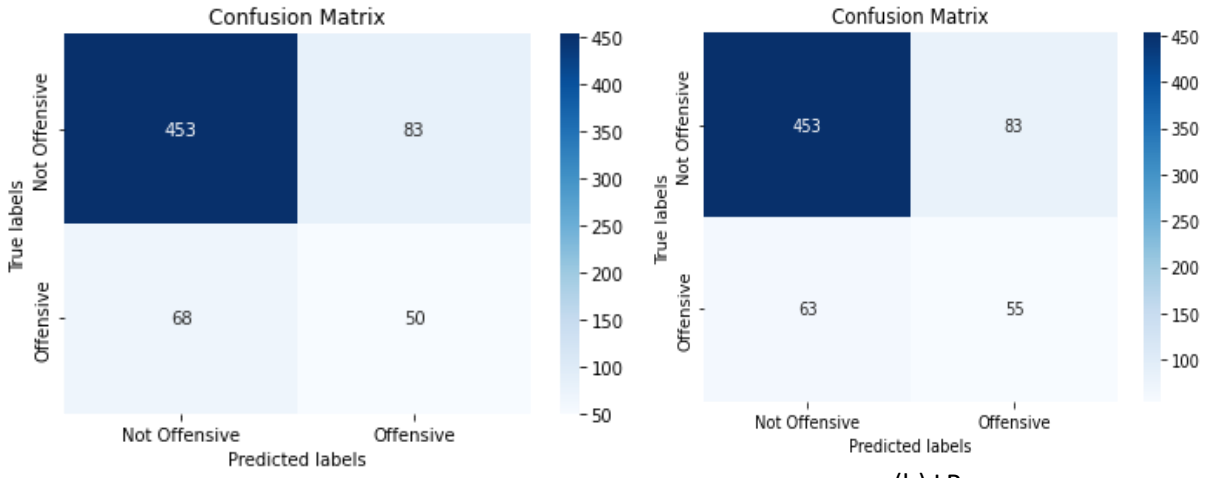
$$\text{Precision} = \text{TP}/(\text{TP} + \text{FP}) \quad (3)$$

F1-Score is defined as the harmonic average of the Precision and Recall, that is, the weighted average of Precision and Recall. It is calculated as in Equation (4).

$$\text{F1-Score} = (2*\text{Precision}*\text{Recall})/(\text{Precision} + \text{Recall}) \quad (4)$$

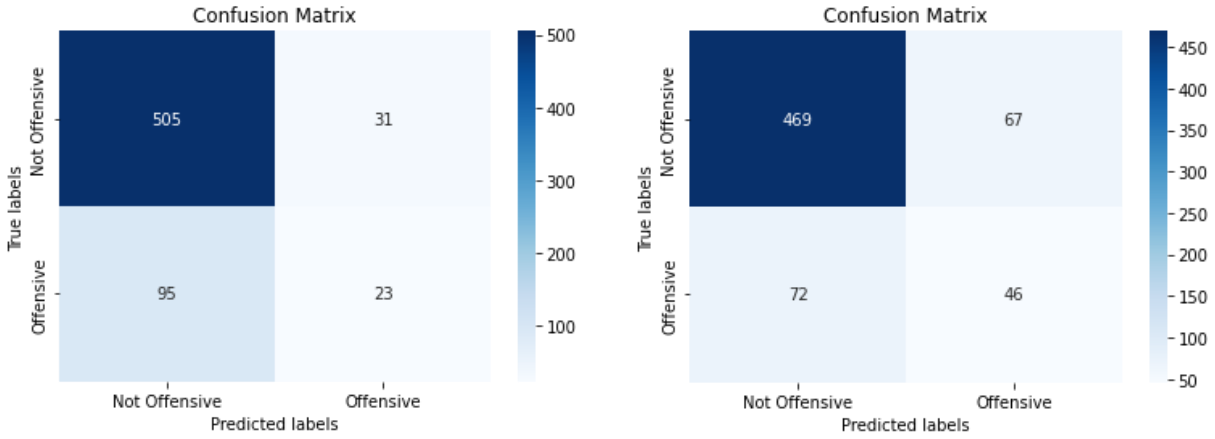
The values of the performance metrics obtained for the test dataset for each of the classifiers are presented in Table 3. Based on the indices, we present the generated confusion matrix, a model matrix that compares the actual target values with those predicted by a machine learning model that helps us perform error analysis of the proposed models. The confusion matrix for the proposed models is shown in Figure 1. The diagonal elements of the confusion matrix represent correct classifications. The predicted classes are represented by X axis, and the actual classes are represented by Y axis. For instance, Figure 1(a) shows that the naïve bayes model classified 50 offensive texts correctly as offensive and 83 offensive texts incorrectly as Not-Offensive.

Table 3 presents the results of classifiers in terms of the performance measures. RoBERTa provides the highest accuracy in comparison to other classifiers. This high accuracy from RoBERTa is because of the significant improvement in the model by training it longer with larger batches over more data, removing the next sentence prediction objective,; training on longer sequences (takes complete sentences as input to the model, as opposed to the base model BERT),; dynamically changing the masking pattern applied to the training data (RoBERTa can make several more distinct masking patterns in the same sequence which reduces the need to significantly increase the number of instances of training) and creating a large vocabulary using Byte-Pair Encoding. However, this model is much more complex and requires a large



(a) Multinomial NB

(b) LR



(c) SVM (d) KNN

Figure 1. Confusion Matrix

amount of time to run. But, [32] recognizes that RoBERTa has a bigger vocabulary, which allows the model to represent any word resulting in more parameters, and the increase in complexity is justified by gain in performance.

Table 3
Performance of classifiers

Classifiers	Class Labels	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Naïve Bayes Multinomial	Not-Offensive	76.911	84.515	86.948	85.714
	Offensive		42.373	37.594	39.841
SVM	Not-Offensive	80.733	94.216	84.167	88.908
	Offensive		19.492	42.593	26.744
KNN	Not-Offensive	78.746	87.500	86.691	87.094
	Offensive		38.983	40.708	39.827
Logistic Regression	Not-Offensive	80.733	84.515	87.791	86.`122
	Offensive		46.610	39.855	42.969
RoBERTa	Not-Offensive	91.390	96.455	84.339	89.991
	Offensive		18.644	53.659	27.673

For offensive class, the Recall and F1-score had low values compared to Not-Offensive classes. We understand that this is due to the smaller number of texts under this class. We have augmented texts for this class using SMOTE techniques, however, the low values for Not-Offensive class persisted. We used count vectors to extract features from the texts in this work - one issue with simple counts is that frequently used words, such as "an" and "the," make their high counts meaningless in the encoded vectors. An alternate option is to use TF-IDF to calculate word frequencies, which might be superior to Count Vectorizers because it not only considers the frequency of words in the corpus but also their importance. Future work can remove the words that are less important for analysis, reducing the input dimensions and making the model building less complex.

5. Conclusion

This paper presented experimental work and respective results of the task to detect offensive content in code-mixed dataset of Dravidian languages to tackle the problem of offensive language in social media. We provided an in-depth review of past techniques in offensive text classification, and new models to automatically detect offensive texts in Tamil. Count vector was used to extract features from the dataset

and different classifiers such as Naïve Bayes Multinomial model, SVM, KNN, Logistic Regression and RoBERTa were built for this task. Of these models, RoBERTa exhibited higher accuracy than other models. Our work could be further extended using other possible numerical/vectorial representation of texts and new classifiers based on neural networks, which have advanced linguistic features. The study contributes to research in offensive text detection for under resourced languages like Tamil, which we hope can inform future work in this area.

References

[1]. J. Blair, "New breed of bullies torment their peers on the Internet," Education Week, 22 (2003): 6.

[2] S.-H. Lee and H.-W. Kim, "Why people post benevolent and malicious comments online," Communications of the ACM, 58(2015) : 74-79.

[3] A. M. Obadimu, "Assessing the Role of Social Media Platforms in the Propagation of Toxicity," University of Arkansas at Little Rock, 2020.

[4] T. De Smedt, S. Jaki, E. Kotzé, L. Saoud, M. Gwóźdź, G. De Pauw, et al., "Multilingual cross-domain perspectives on online hate speech," arXiv preprint arXiv:1809.03944, 2018.

[5] N. Alkiviadou, "Hate speech on social media networks: towards a regulatory framework?," Information & Communications Technology Law, 28 (2019): 19-35.

[6] B. Vandersmissen, "Automated detection of offensive language behavior on social networking sites," IEEE Transaction, 2012, http://lib.ugent.be/catalog/rug01:001887239

[7] Z. Waseem and D. Hovy, Hateful symbols or hateful people? predictive features for hate speech detection on twitter, in: Proceedings of the NAACL student research workshop, (2016):88-93.

[8] M. H. Ribeiro, P. H. Calais, Y. A. Santos, V. A. Almeida, and W. Meira Jr, "Characterizing and detecting hateful users on twitter, in Twelfth international AAAI conference on web and social media, 2018.

[9] R. Kshirsagar, T. Cukuvac, K. McKeown, and S. McGregor, "Predictive embeddings for hate speech detection on twitter," arXiv preprint arXiv:1809.10644, (2018)

[10] P. Mishra, H. Yannakoudakis, and E. Shutova, "Neural character-based composition models for abuse detection," arXiv preprint arXiv:1809.00378,(2018).

[11] J. Mitrović, B. Birkeneder, and M. Granitzer, nlpUP at SemEval-2019 task 6: A deep neural language model for offensive language detection, in: Proceedings of the 13th International Workshop on Semantic Evaluation, 2019.

[12] P. Liu, W. Li, and L. Zou, NULI at SemEval-2019 task 6: Transfer learning for offensive language detection using bidirectional transformers, in: Proceedings of the 13th international workshop on semantic evaluation, (2019): 87-91.

[13] S. D. Swamy, A. Jamatia, and B. Gambäck, "Studying generalisability across abusive language detection datasets," in Proceedings of the 23rd conference on computational natural language learning (CoNLL), 2019, pp. 940-950.

[14] A. Schmidt and M. Wiegand, A survey on hate speech detection using natural language processing, in: Proceedings of the fifth international workshop on natural language processing for social media, 2017, pp. 1-10.

[15] C. Cieri, M. Maxwell, S. Strassel, and J. Tracey, Selection criteria for low resource language programs, in: Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16), 2016, pp. 4543-4549.

[16] B. R. Chakravarthi, R. Priyadharshini, N. Jose, T. Mandl, P. K. Kumaresan, R. Ponnusamy, 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.

[17] S. Suryawanshi and 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, 2021, pp. 126-132.

[18] B. R. Chakravarthi, R. Priyadharshini, S. Banerjee, R. Saldanha, J. P. McCrae, P. Krishnamurthy, et al., Findings of the Shared Task on Machine Translation in Dravidian languages, in: Proceedings of the First Workshop on Speech and Language Technologies for Dravidian Languages, 2021, pp. 119-125.
[19] T. Davidson, D. Warmsley, M. Macy, and I. Weber, Automated hate speech detection and the problem of offensive language, in : Proceedings of the International AAAI Conference on Web and Social Media, 2017.
[20] A. Gaydhani, V. Doma, S. Kendre, and L. Bhagwat, "Detecting hate speech and offensive language on twitter using machine learning: An n-gram and tfidf based approach," arXiv preprint arXiv:1809.08651, 2018.
[21] F. E. Ayo, O. Folorunso, F. T. Ibharalu, and I. A. Osinuga, "Machine learning techniques for hate speech classification of twitter data: State-of-the-art, future challenges and research directions," Computer Science Review, vol. 38, p. 100311, 2020.
[22] S. MacAvaney, H.-R. Yao, E. Yang, K. Russell, N. Goharian, and O. Frieder, "Hate speech detection: Challenges and solutions," PloS one, vol. 14, p. e0221152, 2019.
[23] M. Zampieri, S. Malmasi, P. Nakov, S. Rosenthal, N. Farra, and R. Kumar, "Semeval-2019 task 6: Identifying and categorizing offensive language in social media (offenseval)," arXiv preprint arXiv:1903.08983, 2019.
[24] S. Dowlagar and R. Mamidi, "HASOCOne@ FIRE-HASOC2020: Using BERT and Multilingual BERT models for Hate Speech Detection," arXiv preprint arXiv:2101.09007, 2021.
[25] M. Mozafari, R. Farahbakhsh, and N. Crespi, A BERT-based transfer learning approach for hate speech detection in online social media, in: International Conference on Complex Networks and Their Applications, 2019, pp. 928-940.
[26] Y.-C. Ho and D. L. Pepyne, "Simple explanation of the no-free-lunch theorem and its implications," Journal of optimization theory and applications, 115(2002): 549-570.
[27] B. R. Chakravarthi, R. Sakuntharaj, A. K. Madasamy, S. Thavareesan, and S. C. N. P. B, J. P. McCrae, T. Mandl,, "Overview of the HASOC-DravidianCodeMix Shared Task on Offensive Language Detection in Tamil and Malayalam,," Working Notes of FIRE 2021 - Forum for Information Retrieval Evaluation, CEUR, 2021.
[28] B. R. Chakravarthi and V. Muralidaran, Findings of the shared task on Hope Speech Detection for Equality, Diversity, and Inclusion, in: Proceedings of the First Workshop on Language Technology for Equality, Diversity and Inclusion, 2021, pp. 61-72.
[29] Data Augmentation in NLP. 2021 URL: https://neptune.ai/blog/data-augmentation-nlp
[30] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: synthetic minority over-sampling technique," Journal of artificial intelligence research, 16(2002): 321-357..
[31] S. B. Kotsiantis, I. Zaharakis, and P. Pintelas, "Supervised machine learning: A review of classification techniques," Emerging artificial intelligence applications in computer engineering, 160(2007): 3-24.
[32] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, et al., "Roberta: A robustly optimized bert pretraining approach," arXiv preprint arXiv:1907.11692, 2019.

.