Sentiment Analysis on Multilingual Code-Mixed Kannada Language

Satyam Dutta[1], Himanshi Agrawal[1] and Pradeep Kumar Roy[1]

[1]*Indian Institute of Information Technology, Surat, Gujarat, India*

Abstract

The extended use of the internet and social networking platforms has ushered in a new avenue for people to share their ideas. Sentiment analysis is the process of categorizing people's feelings represented in their thoughts and remarks. It is one of the most debated and researched subjects in Natural Language Processing (NLP) at the moment. Many studies have been offered for sentiment analysis of texts comprising only one language, such as English, Spanish, or Arabic. However, few studies have focused on code-mixed language analysis, which is critical in countries like India, where people speak and express themselves in multiple languages. We present a model in this research, that aids in sentiment analysis of Dravidian Code-Mixed Kannada comments, which achieved a promising weighted F_1-score of 0.66 using the BERT model on the validation dataset, whereas the F1-score on the test dataset was 0.619.

Keywords

Sentiment Analysis, Code-Mixed, Kannada, Machine Learning, Deep Learning, Transformer, BERT

1. Introduction

Sentiment analysis, in the field of natural language processing analyses text, uses computational linguistics and biometrics to uniquely identify, extract, quantify, and study effective states and subjective information [1, 2]. Sentiment analysis is a fascinating yet common field of study. Many from the research community have worked on varying topics and applied analytical tools to get solutions to many problems [3, 4]. Reviews are a vital element of many businesses that deal with products, as they provide with consumers' requirements while also helping many firms develop from the offered feedback [5]. This enables businesses to adapt and benefit more effectively in response to customer demands. However, most of the research conducted on social media posts and YouTube videos comments analysis uses English as the primary language [6].

In terms of culture and languages, India is a land of contrasts. In India, about 447 languages are spoken, with Dravidian languages accounting for 19.64 percent of the population. Anyone may learn and converse in any language. As a result, the internet is brimming with information in a variety of languages and even a combination of languages. So, in a sense, not a lot of research has been conducted on using sentiment analysis for analyzing data in many of the Indian languages, particularly languages from the Dravidian family, namely-Telugu, Tamil, Malayalam, and Kannada [7, 8].

FIRE 2021,Forum for Information Retrieval Evaluation, December 13-17, 2021

satyamdutta2016@gmail.com (S. Dutta); himanshiagrawal26@gmail.com (H. Agrawal); pkroynitp@gmail.com (P. K. Roy)

0000-0003-2372-1567 (S. Dutta); 0000-0002-6943-9498 (H. Agrawal); 0000-0001-5513-2834 (P. K. Roy)

In multilingual communities like India, language mixing, also known as code-mixing, is rather prevalent. People who are multilingual (particularly non-native English speakers), have a tendency to code-mix in their primary language by employing English-based phonetic typing and inserting anglicisms [8]. It is relatively typical to see code-mixing behavior at the word level, in addition to mixing languages at the sentence level. These phenomena present a considerable barrier to traditional NLP systems, which rely on monolingual resources to process the inputs. NLP tasks such as language recognition and translation [9], speech tagging, parsing and semantic analysis and processing, offensive language detection [10, 11] are all affected by code mixing. Traditional NLP systems rely extensively on one language, which limits their ability to handle challenges like English-based phonetic typing, word-level code-mixing, and other issues while dealing with code-mixed text.

The focus of this study is to bring the challenge of sentiment analysis in a code-mixed textual data to the attention of the research community. The data for the "Dravidian-CodeMix-FIRE 2021[1]" challenge was scraped from the YouTube comments, where each instance of data is labeled with one of the sentiment polarities: "positive, negative, neutral, mixed feelings or not in the intended languages" [8]. We have described our numerous Machine Learning and Deep Learning methodologies in this work, as well as our final proposed model, which is based on Transformers, Bidirectional Encoder Representations from Transformers (BERT model).

The rest of the paper is organized as follows: Section 2 discusses the related works. Section 3 discusses the proposed methodology. Section 4 discusses the experimental outcomes of various machine learning and deep learning models. Finally, Section 5 concludes the work.

2. Literature Review

Sentiment analysis plays a major role in the decision-making system and is widely used in various filed including recommendation systems, e-commerce, hotel business and many others. The importance of the sentiment analysis in different domains attracted researchers to build an automated system, and hence many models have been reported in recent years [12, 13, 14]. Ouyang et al. [14] proposed a deep learning-based framework for sentiment analysis. They used the word2Vec technique for text to a vector representation. Then three layers of CNN followed by a pooling layer were used for the sentiment analysis task. Li et al. [15] developed a model using parallel CNN and LSTM network for sentiment analysis on English movie reviews and Chinese tourism reviews dataset. The sentiment padding technique was developed by the authors and claimed the technique was better than zero padding. Further, lexicon integrated CNN and Bi-LSTM model was developed. Ombabi et al. [16] used one layer of CNN and a two-layered LSTM model for Arabic sentiment analysis purposes. The features extracted by CNN and LSTM model were given to the SVM model to predict the sentiment. Authors used FastText embedding for text representation and achieved 90.75% accuracy on multi-domain corpus for the best case.

The major issues with the existing researches including the language dependency of the model. Mainly English, Chinese, Arabic or others, but a single language dataset was used for model development, which can not process the multilingual data. The existing models

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

are not considered for evaluation if the message, review, or tweets consist of more than one language like English-Hindi, English-Chinese, English-Tamil, and similar. However, as per need, few recent works reported the deep learning models to address these issues recently [2, 17]. They developed the Dravidian code-mixed dataset and developed multiple machines and deep learning frameworks to handle the code-mixed input data. The shortcomings of the existing techniques and limited research on Dravidian code-mixed data motivated us for this work.

3. Methodology

All the applied Machine Learning, Deep Learning, and Transformer models are described in depth in this section, the developed code for this work can be found on the given link[2]. On the Kannada dataset [18], we first used several basic Machine Learning models like Random Forest, Support Vector Machine, and Logistic Regression with default values of hyper-parameters. The performances of these models were evaluated in terms of precision, recall and F_1-score [19]. Table 1 shows the data statistics that were used in the analysis. The working steps of the proposed methodology are shown in Figure 1. The dataset consisted of a large number of invalid contents like emoji, URLs, and others. Hence, data cleaning and the prepossessing task is performed prior to developing the model, such as eliminating all the emojis, emoticons, and special symbols. We also removed all of the numbers and transformed the data to lowercase. For text encoding, we employed the n-gram Term Frequency-Inverse Document Frequency vectorizer.

We created a hybrid model combining Convolutional Neural Networks (CNN) [20, 21] and Bidirectional Long Short-Term Memory (Bi-LSTM) networks after the machine learning models failed to produce promising results. The hybrid model's details, as well as their hyperparameters, are listed in Table 2. This hybrid model outperformed the machine learning models by a little margin. However, to attain even better results, we built a Deep Learning model by utilizing the Transformer architecture, known as Bidirectional Encoder Representations from Transformers (BERT) [22]. We used two different methods to implement this model:
1. BERT model implementation from scratch using TensorFlow[3].
2. BERT model implementation using a wrapper known as ktrain[4].

The basic difference between these methods is, in the first one, we must manually reformat the data in ways that the BERT model accepts. Whereas, we only need to specify the hyperparameters value in the ktrain [23] method, and ktrain will take care of the rest. Both strategies use a similar implementation and indicate the same that transfer learning is the preferred method for downstream jobs. Nonetheless, both strategies are discussed here because we began with the conventional BERT model, where we built our model using BERT from the ground up to better understand how it works. After knowing the internal workings of BERT, we went for a strategy to create the same model architecture with less effort in less time, avoiding most of the pre-processing steps. The following subsections provide an explanation of the two strategies mentioned above.

[2]https://github.com/RogNet11/Sentiment-Analysis-of-Code-Mixed-Dravidian-Text
[3]https://www.tensorflow.org/text/tutorials/classify_text_with_bert
[4]https://github.com/amaiya/ktrain

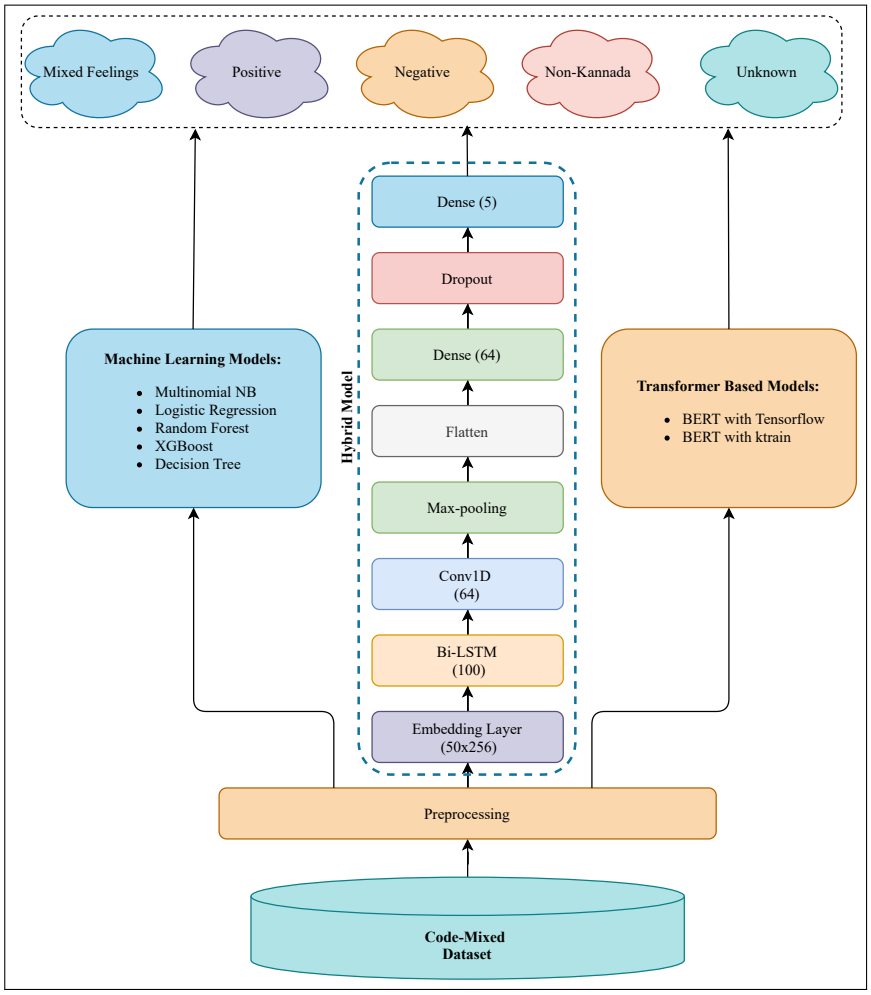

Figure 1: Framework for Sentiment Analysis for Code-Mixed dataset

3.1. BERT from scratch using TensorFlow

BERT is a pre-trained transformer-based model. In addition to standard text input, the BERT employs a feature known as 'Attention Masks,' which aids in the better capture of phrase context. The research article [24] provides a full explanation about it.

A transformer consists of an encoder that reads the text input and a decoder that generates a prediction for the input. Nevertheless, only the encoder mechanism is required here because BERT aims to create a language model. Compared to the directional models that read the input text sequentially, the Transformer encoder's most crucial characteristic is that it reads the complete sequence of words at once. This property enables the model to deduce the context of a word from its surroundings. It is faster too because it does not follow the recursive structure of RNNs and LSTMs [25], but rather parallelization.

Table 1
Data Statistics for Code-Mixed Kannada Dataset

Classes	Training	Development
Positive	2823	321
Negative	1188	139
not-Kannada	916	110
unknown state	711	69
Mixed feelings	574	52
Total	6212	691

We employed the 'bert-base-uncased'[5] model for this study, which produced better outcomes than the Roberta and XLNet models. We also tried a few other multilingual models, but their accuracy was lower than the one we used since, although being multilingual, the other models are trained on monolingual phonetics, whereas the dataset largely contains code mixed inputs. Each BERT model has its own tokenizer, which we used to encode the input data into vectors and attention masks. After that, we loaded the data arrays into a TensorFlow dataset object and turned them into TensorFlow tuples. Then, we constructed the model with the hyperparameters listed in Table 3.

3.2. BERT with ktrain

ktrain is a lightweight wrapper for TensorFlow, Keras and other libraries, making it much easier to design, train, and deploy neural networks and other machine learning models [23]. It comes with all of the methods and pre-processing procedures pre-programmed, and it is simple to specify and fine-tune the parameters to generate a good model. A list of the parameters supplied to ktrain is presented in Table 4.

When comparing the hyperparameters of both implementations, the only variation is the number of epochs, as both approaches are nearly identical apart from implementation. However, because the method 'early stopping' is utilized, the number of epochs in run time will be the same. Different learning rates were also tried, with the one that produced the best results being chosen. During the testing of various language models, some of the models caused problems when implemented from the scratch. As a result, ktrain was used to conduct the experiments and obtain the final data, however both methods yielded similar results. With ktrain also, the same model was used: 'bert-base-uncased'.

4. Experimental Results

In the task of Dravidian-CodeMix-FIRE2021, we have classified code mixed Kannada social media comments into five different sentiment classes: (i) Mixed feelings, (ii) Positive, (iii) Negative, (iv) Not related to that language (Not-Kannada) and (v) Unknown state. Table 2 shows Hyper-parameters details for the proposed hybrid model for Kannada Dataset. Table

[5]https://github.com/google-research/bert

Table 2
Hyperparameter details for the proposed hybrid model for Kannada Dataset

Hyper-parameters	CNN + Bi-LSTM model
Number of Bi-LSTM layers	1
Number of CNN layers	1
Embedding layer dimensions	50x256
Pooling Layer	MaxPooling1D
Pooling window Size	2
Dropout rate	0.2
Activation functions	ReLU, Softmax
Epochs	5
Loss	Categorical Crossentropy
Optimizer	Adam

Table 3
Hyperparameter details for the BERT model from scratch with TensorFlow for Kannada Dataset

Hyper-parameters	BERT model
Maximum length of comments	64
Number of BERT layers	1
Batch size	64
Pooling Layer	MaxPooling1D
Pooling window Size	2
Dropout rate	0.2
Activation functions	ReLU, Softmax
Epochs	15
Loss	Categorical Crossentropy
Optimizer	Adam
Learning rate	1e-4

Table 4
Hyperparameters details for the BERT model with ktrain for Kannada Dataset

Hyper-parameters	BERT model
Maximum length of comments	64
Batch size	64
Preprocess mode	bert
Epochs	5
Learning rate	1e-4

3 shows Hyper-parameters details for the BERT model from scratch with TensorFlow for Kannada Dataset, and Table 4 shows hyperparameter details for the BERT model with ktrain for Kannada Dataset. Table 5 shows results for code-mixed Kannada sentiment analysis using various Machine Learning models. Support vector machine achieved the highest accuracy of 0.56 and weighted F_1-score of 0.50 compared to other machine learning models. Table 6 shows

Table 5
Results for code-mixed Kannada sentiment analysis using various Machine Learning Models on validation data

Models	Accuracy	Precision	Recall	Weighted F_1-score
Multinomial Naïve Bayes	0.51	0.61	0.51	0.40
Logistic Regression	0.55	0.61	0.55	0.47
Random Forest	0.45	0.57	0.45	0.45
XGBoost	0.50	0.55	0.50	0.40
Decision Tree	0.41	0.52	0.41	0.42
Support Vector Machine	**0.56**	**0.58**	**0.56**	**0.50**

Table 6
Results for code-mixed Kannada sentiment analysis using hybrid Deep Learning Model on validation data

Classes	Precision	Recall	F_1-score
Negative	0.00	0.00	0.00
Positive	0.68	0.37	0.48
not-Kannada	0.71	0.75	0.73
Mixed feelings	0.75	0.52	0.61
unknown state	0.70	0.10	0.18
weighted avg	**0.66**	**0.52**	**0.55**

Table 7
Results for code-mixed Kannada sentiment analysis using BERT on validation data

	BERT with TensorFlow			BERT with ktrain		
Classes	Precision	Recall	F_1-score	Precision	Recall	F_1-score
Negative	0.23	0.13	0.17	0.26	0.31	0.28
Positive	0.66	0.63	0.64	0.68	0.60	0.64
not-Kannada	0.72	0.80	0.76	0.72	0.79	0.75
Mixed feelings	0.60	0.72	0.65	0.70	0.64	0.67
unknown state	0.56	0.33	0.42	0.58	0.43	0.50
weighted avg	**0.64**	**0.66**	**0.64**	**0.66**	**0.66**	**0.66**

code-mixed Kannada sentiment analysis results using hybrid Deep Learning Model(Bi-LSTM + CNN), which was improved from machine learning models and provided promising weighted precision of 0.66, recall of 0.52 and F_1-score of 0.55.

Table 7 shows results for code-mixed Kannada sentiment analysis using “BERT with TensorFlow” and “BERT with ktrain”. “BERT with TensorFlow” model achieved the weighted precision value of 0.64, recall of 0.66, and F_1-score of 0.64, whereas, “BERT with ktrain" model achieved 0.66, 0.66, 0.66 as weighted precision, recall and F_1-score respectively. The above-mentioned results are obtained on the validation dataset, that was provided to us by the organizers. Our best

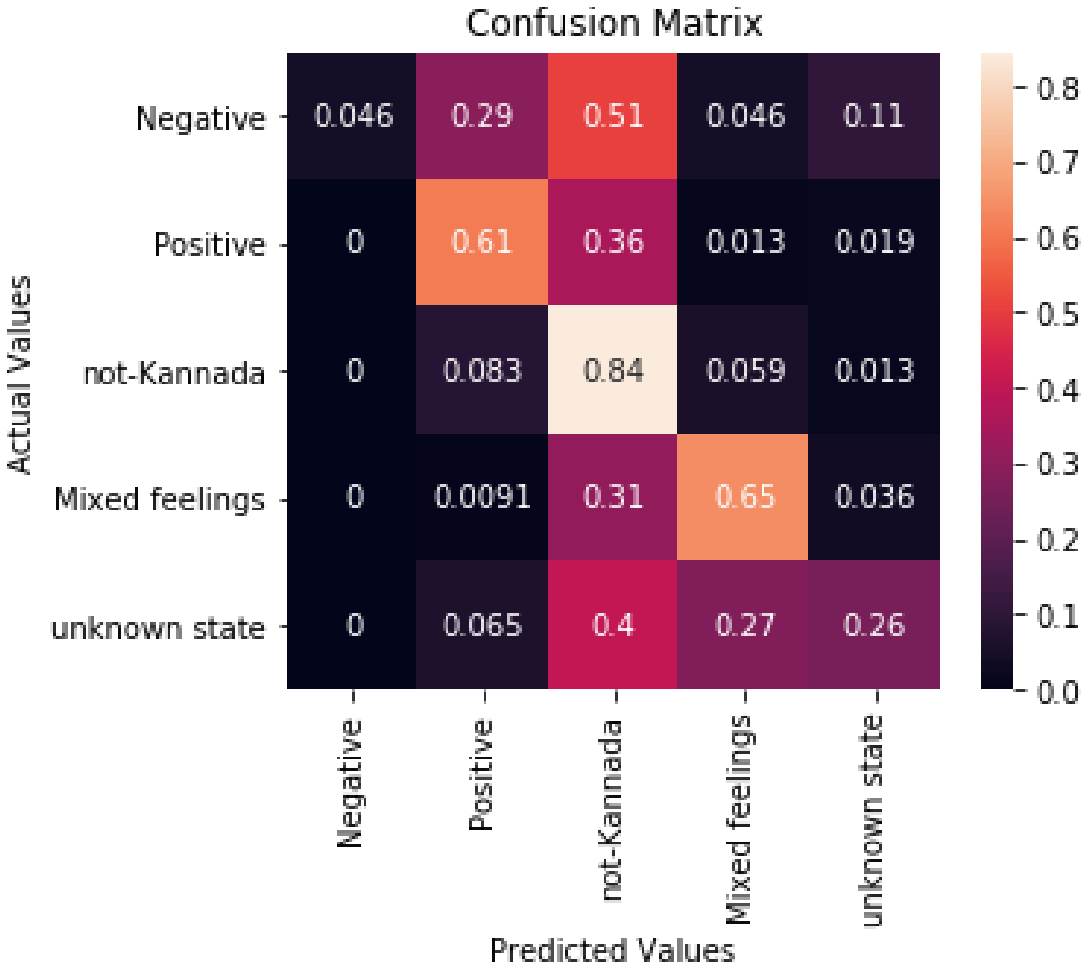

Figure 2: Confusion Matrix for BERT with ktrain for Kannada code-mixed dataset

model, i.e., "BERT with ktrain" achieved the weighted precision, recall and F_1-scores as 0.672, 0.654 and 0.619 respectively on the test dataset. To further analyze the misclassified instances from the various classes, the confusion matrix obtained using the best performing model, is plotted as shown in Figure 2. Among all, 51% of negative, 36% of positive, 31% mixed-feelings, and 40% of unknown state are misclassified to not-Kannada category, which may be the reason behind lower overall weighted F_1-score. Data prepossessing and more samples of the data in another category might be needed to improve the overall prediction accuracy.

5. Conclusion

Sentiment analysis has been the subject of discussion because it is seen to be significant in light of the growing amount of data available on the internet. To classify our code-mixed data into distinct sentiment categories, we used a variety of machine learning, deep learning and transformer-based models, with the transformer-based BERT model outperforming the other variants. The BERT model was implemented in two ways: one from the ground up and another that is advanced and simple. For the code-mixed Kannada dataset, latter obtained a promising weighted F_1-score of 0.66 on the validation dataset; while on the test dataset, the weighted F_1-score achieved was 0.619.

References

[1] A. Kumar, S. Saumya, J. P. Singh, Nitp-ai-nlp@ Dravidian-codemix-fire2020: A hybrid cnn and bi-lstm network for sentiment analysis of Dravidian code-mixed social media posts., in: FIRE (Working Notes), 2020, pp. 582–590.

[2] B. R. Chakravarthi, R. Priyadharshini, V. Muralidaran, S. Suryawanshi, N. Jose, E. Sherly, J. P. McCrae, Overview of the track on sentiment analysis for Dravidian languages in code-mixed text, in: Forum for Information Retrieval Evaluation, 2020, pp. 21–24.

[3] B. R. Chakravarthi, R. Priyadharshini, S. Thavareesan, D. Chinnappa, T. Durairaj, 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 2021, in: Working Notes of FIRE 2021 - Forum for Information Retrieval Evaluation, CEUR, 2021.

[4] 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://aclanthology.org/2020.sltu-1.25.

[5] S. Saumya, J. P. Singh, Detection of spam reviews: a sentiment analysis approach, Csi Transactions on ICT 6 (2018) 137–148.

[6] S. Saumya, A. K. Mishra, Iiit_dwd@ lt-edi-eacl2021: Hope speech detection in youtube multilingual comments, in: Proceedings of the First Workshop on Language Technology for Equality, Diversity and Inclusion, 2021, pp. 107–113.

[7] 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.

[8] R. Priyadharshini, B. R. Chakravarthi, S. Thavareesan, D. Chinnappa, D. Thenmozhi, R. Ponnusamy, 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.

[9] B. R. Chakravarthi, R. Priyadharshini, S. Banerjee, R. Saldanha, J. P. McCrae, A. K. M, P. Krishnamurthy, M. Johnson, 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, Association for Computational Linguistics, Kyiv, 2021, pp. 119–125. URL: https://aclanthology.org/2021.dravidianlangtech-1.15.

[10] A. Hande, K. Puranik, K. Yasaswini, R. Priyadharshini, S. Thavareesan, A. Sampath, K. Shanmugavadivel, D. Thenmozhi, B. R. Chakravarthi, Offensive language identification in low-resourced code-mixed Dravidian languages using pseudo-labeling, 2021. arXiv:2108.12177.

[11] 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.

[12] B. Liu, et al., Sentiment analysis and subjectivity., Handbook of natural language processing 2 (2010) 627–666.

[13] R. Feldman, Techniques and applications for sentiment analysis, Communications of the ACM 56 (2013) 82–89.

[14] X. Ouyang, P. Zhou, C. H. Li, L. Liu, Sentiment analysis using convolutional neural network, in: 2015 IEEE international conference on computer and information technology; ubiquitous computing and communications; dependable, autonomic and secure computing; pervasive intelligence and computing, IEEE, 2015, pp. 2359–2364.

[15] W. Li, L. Zhu, Y. Shi, K. Guo, E. Cambria, User reviews: Sentiment analysis using lexicon integrated two-channel cnn–lstm family models, Applied Soft Computing 94 (2020) 106435.

[16] A. H. Ombabi, W. Ouarda, A. M. Alimi, Deep learning cnn–lstm framework for arabic sentiment analysis using textual information shared in social networks, Social Network Analysis and Mining 10 (2020) 1–13.

[17] 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://aclanthology.org/2020.sltu-1.28.

[18] 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://aclanthology.org/2020.peoples-1.6.

[19] 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.

[20] 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.

[21] P. K. Roy, Multilayer convolutional neural network to filter low quality content from quora, Neural Processing Letters 52 (2020) 805–821.
[22] 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).
[23] A. S. Maiya, ktrain: A low-code library for augmented machine learning, arXiv preprint arXiv:2004.10703 (2020).
[24] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, in: Advances in neural information processing systems, 2017, pp. 5998–6008.
[25] P. K. Roy, Deep neural network to predict answer votes on community question answering sites, Neural Processing Letters 53 (2021) 1633–1646.