Sentiment Analysis of Dravidian-CodeMix Language

Ankit Kumar Mishra[1], Sunil Saumya[2] and Abhinav Kumar[1]

[1]Goa University, Goa, India
[2]Indian Institute of Information Technology Dharwad, Karnataka, India
[1]Siksha 'O' Anusandhan Deemed to be University, Bhubaneswar, India

Abstract

The computational examination of people's opinions, attitudes, and emotions conveyed in written language is known as sentiment analysis or opinion mining. In recent years, it has become one of the most active study fields in natural language processing and text mining. Sentiment analysis of social media texts, which are predominantly code-mixed for Dravidian languages, is becoming more popular. In a multilingual community, code-mixing is common, and code-mixed writings are produced using native and non-native scripts. The current paper uses machine learning, deep learning, and parallel hybrid deep earning models to identify sentiments in Dravidian code-mixed social media text. The experiments were conducted using a dataset from the *Dravidian-CodeMix-FIRE 2021*[1] competition, which included YouTube comments in Tamil, Malayalam, and Kannada code-mixed languages.

Keywords

Sentiment Analysis, code-mixed, deep learning, machine learning, Dravidian languages, CEUR-WS

1. Introduction

Human language is a difficult beast to grasp. It is tough to teach a machine to recognize the different linguistic nuances, cultural variances, slang, and misspellings that appear in social media discussions. It's considerably more difficult to teach a machine to recognize how context affects tone. When it comes to interpreting the tone of a piece of literature, humans are fairly intuitive. Consider this sentence: *"I love waiting for the doctor"*. Most people would immediately recognize that the preceding sentence is mocking. We understand that everyone must wait in line to see a doctor, but we all despise waiting. We can easily identify the sentiment as negative by using this contextual information in the statement. However, a machine reading the statement above might recognize the word "love" and categorize it as positive without knowing the context.

The computational examination of people's opinions, sentiments, attitudes, and emotions conveyed in written language is known as sentiment analysis or opinion mining [1]. Due to its vast range of applications, it has become one of the most active research domains in natural language processing and text mining in recent years [2]. Opinions influence our behavior and are crucial to almost all human activity [3]. Whenever we need to make a decision, we want

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

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

ankitmishra.co.in@gmail.com (A. K. Mishra); sunil.saumya@iiitdwd.ac.in (S. Saumya); abhinavanand05@gmail.com (A. Kumar)

to hear other people's perspectives. For example, if we want to buy a product from an online platform or a physical store, we first gather information about it, such as reading peer buyer reviews and viewing videos, to determine whether the product is right for us [4, 5]. Due to the massive amount of data available on the internet in various forms, manually identifying sentiments is nearly impossible. As a result, a trained machine is required to read the text and automatically return its sentiment [6, 7].

The majority of sentiment analysis research has taken place in the last one or two decades [8, 9, 10, 11, 12, 13, 14, 15, 16]. The cause for this could be the shift in our business from physical to virtual platforms [17]. The participation of social media and e-commerce among virtual platforms has compelled us to build a variety of sentiment analysis solutions for distinct circumstances [18]. Because of its importance to business and society as a whole, sentiment analysis research has moved beyond computer science to management sciences and social sciences.

Recently, there has been a surge in interest in sentiment analysis of social media posts and reviews, which are frequently expressed in code-mixed formats [19, 20, 21]. In a multilingual population, code-mixing is a common occurrence. People in a multilingual country like India generally use code-mixed discourse in both physical and online settings. Code-mixed texts are frequently written in a single script and combine two languages (for example, Hindi and English). Code-mixing is demonstrated in the following sentence: "Maine aaj tak itna swadist bhojan nahi kiya." It is a mixed-code Hindi-English sentence. Even though it is written in English script, many individuals who do not speak Hindi will be unable to comprehend the sentiment/meaning of this line. Identifying sentiments in multilingual settings is tough even for machines that have been trained for monolingual situations.

The current study analyzes the emotion of YouTube comments in code-mixed Dravidian languages like "Tamil-English ", "Malayalam-English," and "Kannada-English". The dataset used in this work is a part of the task proposed in "Dravidian-CodeMix-FIRE 2021 task". The current paper develops various classifiers to perform message-level classification of each YouTube comment into one of the following classes that are "Positive", "negative", "not-tamil/malayalam/kannada", "unknown_state" and "mixed-feelings". The current paper examines the robustness of several conventional machine learning models (such as support vector machine, random forest, and decision tree) and deep learning models in serial settings (such as convolutional neural network (CNN), long short term memory(LSTM), bidirectional-long short term memory (Bi-LSTM), and gated recurrent unit networks (GRU)) and parallel settings (two parallel CNNs, or two parallel LSTMs, two parallel CNN and LSTM, three parallel LSTMs) for the sentiment analysis task. The experimental results on a parallel hybrid model comprising LSTM networks reported the best accuracy of 0.56 F1-score.

The organization of the paper is as follows: the methodology proposed in the paper and dataset statistics are described in Section 2. The results of various experiments are explained in Section 3. The current paper summarizes the most important findings of the paper in Section 4.

Language	Positive		Negative		unknown state		mixed feelings		Not in intended language		Total	
	Train	Dev	Train	Dev	Train	Dev	Train	Dev	Train	Dev	Train	Dev
Tamil	20070	2257	4271	480	5628	611	4020	438	1667	176	35656	3962
Malayalam	6421	706	2105	237	5279	580	926	102	1157	141	15888	1766
Kannada	2823	321	1188	139	711	69	574	52	916	110	6212	691

Table 1
Train and Dev datasets description

2. Methodology

The current research proposes a multi-task classification model for sentiment analysis of Dravidian code-mixed YouTube comments. Figure 1 depicts a thorough flow of the proposed methods. Three parallel layers of LSTM networks are concatenated to extract features from YouTube comments, as shown in Figure 1. The output layer receives the concatenated vector as input for multi-task classification.

2.1. Task and Data Description

The task aimed to classify each YouTube comment into one of the five classes that are “Positive”, “negative”, “not-tamil/malayalam/kannada”, “unknown_state” and “mixed-feelings”.

The competition datasets were made available in stages. Initially, training and development data for each Tamil, Malayalam, and Kannada corpus were released separately, resulting in six separate data files. Each file contained two fields: a text field and a category field. Except for a few situations where it was more than one, the average comment length in a corpus of Tamil, Malayalam, and Kannada was one. Table 1 contains descriptions of both the training and development sets for all three corpora. The present system was trained on 35656, 15888, and 6212 samples for Tamil, Malayalam, and Kannada, respectively, and validated on 3962, 1766, and 691 samples for Tamil, Malayalam, and Kannada, as given in Table 1. The organizers later provided the test dataset, based on which the final ranking of submitted models was determined.

2.2. Data Preprocessing

Every language dataset (train, development, and test sets) was subjected to the preprocessing steps. Initially, all punctuation was removed from the texts and they were changed to lowercase. Certain rows have only one word with no apparent meaning, such as “Suppperrrrrrrrrrrrrr” that was eliminated. Sentences with two or fewer letters were removed because they had little impact on the dataset. Finally, the training dataset was prepared. After that, the cleaned text was tokenized and encoded into a series of token indexes. Finally, padding with a maximum length of 100 was used to ensure that all texts were of similar length.

2.3. Classification models

For sentiment analysis of YouTube comments, the current work used different classification models. This section covers the many conventional classifiers, deep learning classifiers, and

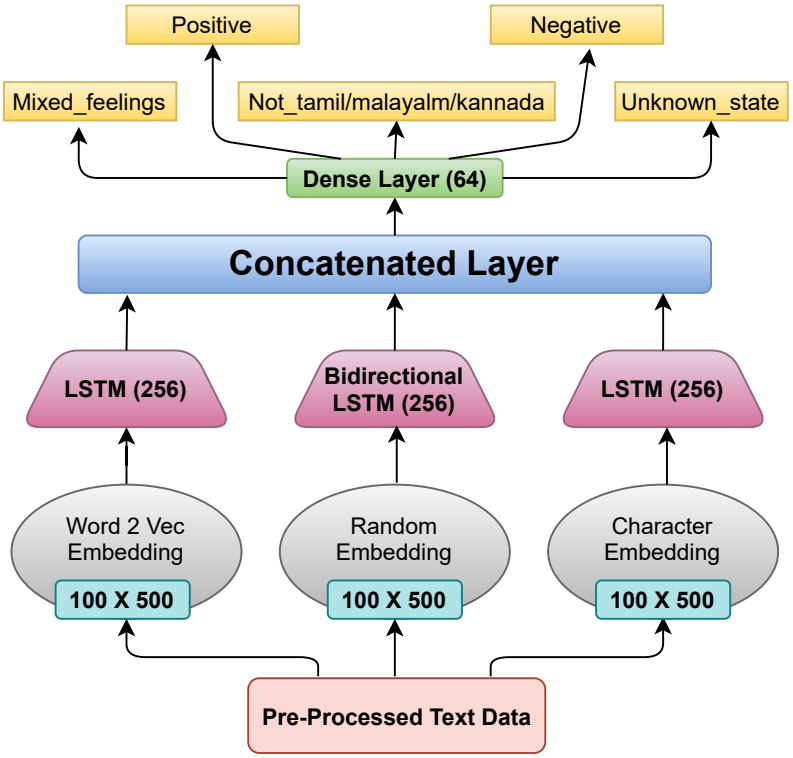

Figure 1: The proposed model flow diagram

hybrid classifiers that were utilized in the study.

2.3.1. Conventional Machine Learning Classifiers

The Support Vector Machine (SVM), Logistic Regression (LR), and Random Forest (RF) are three traditional machine learning-based models that were developed to categorize YouTube comments. Tf-idf vectors constructed from Tamil, Malayalam, and Kannada comments were used as input to these classifiers. Initially, the library *WhiteSpace Tokenizer* was used to tokenize the comments. Further, the *Porter stemmer* library was used to stem the tokenized text. Finally, a *Tf-idf vectorizer* was used to vectorize the stemmed text. The Tf-idf vector-based machine learning models had extremely long training times.

2.3.2. Deep learning classifier

The current study used convolutional neural networks (CNN) and recurrent neural network versions such as long short term memory (LSTM), Bidirectional long short term memory (BiLSTM), and Gated recurrent units (GRU) in single and multiple layers in the category of deep networks. The tokenized texts were first encoded and then padded with maximum comment length 100, 60, and 50 for Tamil, Malayalam, and Kannada datasets, respectively.

The input to the above deep models was a one-hot vector that represented each word in a

language in its vocabulary dimensions. For example, the vocabulary size in the Tamil corpus was 55810, therefore each word in the Tamil data was represented by a vector with 55810 dimensions (1×55810). The one-hot input representation was a high-dimensional sparse vector with a single 1' (representing the token's index) and all zeros. This high dimensional space vector was reduced into low dimensional dense value vector before passing it as an input to the deep models. An embedding layer that represented every word in a 500 dimensional (1×500) dense vector was utilized to turn one hot vector into a low dimensional dense valued vector.

While representing one-hot vector into embedded dense vector several pre-trained weights from Word2Vec and random embeddings were used. The obtained embedded vectors were fed to the deep networks (such as single and multiple layers of CNN, LSTM, GRU, and Bi-LSTM) for further steps like contextual feature extraction followed by classification.

2.3.3. Hybrid Network

Several hybrid deep neural network configurations, such as parallel CNN-CNN, CNN-LSTM, CNN-BiLSTM, CNN-CNN-CNN, CNN-LSTM-BiLSTM, CNN-CNN-GRU, CNN-LSTM-BiLSTM, and so on, have also been developed for sentiment analysis tasks. Figure 1 shows the architecture of the best hybrid model. The embedding vectors were used as the input to these networks, as indicated in Section 2.3.2.

For all of the datasets, the model shown in Figure 1 reported the best accuracy. The number of words fed from the pre-processed text for Tamil, Malayalam, and Kannada, respectively, was 100, 60, and 50. Every word in Tamil (with a dimension of 1×55810), Malayalam (with a dimension of 1×34012), Kannada (with a dimension of 1×13015), and every character in Tamil (with a dimension of 1×358), Malayalam (with a dimension of 1×230), and Kannada (with a dimension of 1×222) was represented in 500 dimensional embedded vector (1×500) using word2vec, random embedding, and character embeddings. The parallel Bidirectional long short-term memory (BiLSTM) models were fed the embedded vectors. The retrieved features (or output) from the three parallel layers were then combined into a single vector. The concatenated vector was then fed to two serial dense layers having neurons 64 and 5 respectively. The last dense layer (having 5 neurons shown as output labels in Figure 1) is used to classify each comment into one of five categories of sentiment. For all datasets, the model was trained for "50" epoch using batch size "64" with the "*Adam*" optimizer and "*categorical cross entropy*" loss function.

3. Results

As discussed above several classification models were built and experimented with the given Dravidian codemixed datasets. We are reporting the results of the submitted models in the competition and a few other comparable models. All models were built in *Python* using libraries *Sklearn*, *Keras*, *Pandas*, and *Numpy*. The metric used to evaluate the performance of a model was the weighted F1-score (or weighted F1). Table 2 shows the experimental results of various machine learning, deep learning, and hybrid learning models on the development dataset. In conventional classifiers, SVM, Decision tree and Random forest performances are shown in Table 2. The feature used for conventional classifiers was Tf-idf vector whereas, for deep and hybrid models, features were embeddings.

Language	Type	Model	Embedding/Feature	Dev weighted F1
Tamil	**Conventional Learning models**	SVM	Tf-idf	0.45
		Random Forest	Tf-idf	0.46
		Decision Tree	Tf-idf	0.45
	Deep learning model	LSTM	Random word	0.45
		CNN	Random word	0.45
		LSTM	Word2vec	0.51
		CNN	Word2vec	0.50
		BiLSTM	Random word	0.46
		BiLSTM	Word2vec	0.52
		CNN	Random char	0.42
		LSTM	Random char	0.46
		BiLSTM	Random char	0.50
		GRU	Word2vec/Random word	0.50
	Hybrid learning models	2 parallel CNN	Word2vec, Random word	0.51
		2 parallel CNN & LSTM	Word2vec & Random word	0.51
		2 parallel LSTM	Word2vec & Random word	0.52
		2 parallel LSTM & Bi-LSTM	Word2vec & Random word	0.53
		3 parallel CNN	Word2vec, random char & Random word	0.50
		3 parallel LSTM	Word2vec, random char & Random word	0.51
		3 parallel Bi-LSTM	**Word2vec, random char & Random word**	**0.55**
		3 parallel CNN & LSTM	Word2vec, random char & Random word	0.52
		3 parallel LSTM, CNN & Bi-LSTM	Word2vec,random char & random word	0.53
		2 parallel GRU	Random word/word2vec	0.50
Malayalam	**Conventional learning models**	SVM	Tf-idf	0.56
		Random Forest	Tf-idf	0.59
		Decision Tree	Tf-idf	0.57
	Deep Learning models	CNN	Word2vec	0.60
		CNN	Random word	0.60
		CNN	Random char	0.58
		LSTM	word2vec	0.62
		LSTM	Random word	0.61
		LSTM	Random char	0.56
		GRU	word2vec	0.60
		GRU	random word	0.62
		Bi-LSTM	word2vec	0.62
		Bi-LSTM	random word	0.61
		Bi-LSTM	random char	0.55
	Hybrid Models	2-parallel CNN	word2vec & random	0.61
		2-parallel CNN & LSTM	word2vec & random word	0.61
		2-parallel LSTM	word2vec & random word	0.62
		2-parallel LSTM & Bi-LSTM	word2vec & random word	0.62
		3-parallel CNN	word2vec, random word & random char	0.60
		3-parallel LSTM	word2vec, random word & random char	0.61
		3-parallel Bi-LSTM	**word2vec, random word & random char**	**0.63**
		3-parallel CNN & LSTM	word2vec, random word & random char	0.62
		3-parallel LSTM, CNN and Bi-LSTM	word2vec, random word & random char	0.61
		3-parallel GRU, CNN and Bi-LSTM	word2vec, random word & random char	0.62
Kannada	**conventional learning model**	SVM	Tf-idf	0.46
		Random Forest	Tf-idf	0.50
		Decision Tree	Tf-idf	0.47
	Deep Learning Model	CNN	word2vec	0.51
		LSTM	word2vec	0.51
		Bi-LSTM	word2vec	0.52
		Bi-LSTM	random word	0.52
		CNN	random word	0.51
		LSTM	random word	0.51
		Bi-LSTM	random char	0.54
		LSTM	random char	0.53
		GRU	word2vec	0.51
		GRU	random char	0.51
		CNN	random char	0.51
	Hybrid Learning Model	2-parallel CNN	word2vec & random word	0.50
		2 parallel CNN & LSTM	random word & word2vec	0.52
		2-parallel LSTM & GRU	random word & word2vec	0.53
		2-parallel LSTM	random word & word2vec	0.52
		2–parallel Bi-LSTM	random word & word2vec	0.54
		3-parallel CNN	random word, word2vec & random char	0.51
		3-parallel LSTM	random word, word2vec & random char	0.53
		3-parallel Bi-lSTM	**random word, word2vec & random char**	**0.56**
		3-parallel CNN & Bi-LSTM	random word, word2vec & random char	0.55
		3-parallel LSTM & Bi-LSTM	random word, word2vec & random char	0.55
		3-parallel CNN, LSTM & Bi-LSTM	word2vec, random char & random word	0.54

Table 2
Development dataset results of various models

Language	Model	Embedding/Features	Dev weighted F1	Test Weighted F1	Ranking
Tamil	3-parallel Bi-LSTM	Word2vec, Random word & Random char	0.55	0.555	15
Malayalam	3-Parallel Bi-LSTM	Word2vec. Random word & Random char	0.63	0.616	12
Kannada	3-Parallel Bi-LSTM	Word2vec, Random word & Random char	0.56	0.553	12

Table 3
Development and test dataset results of best performing submitted models

In conventional models, for the Tamil dataset, SVM reported weighted F1 0.45, Random Forest reported weighted F1 0.46 and Decision tree reported weighted F1 0.45. The performance of the traditional classifier was poor as compared to deep learning classifiers as it is shown in Table 2. The LSTM and Bi-LSTM models with word2vec embedding reported 0.51 and 0.52 weighted F1 scores, but at the same time, the performance of the CNN model with word2vec embedding was very low as weighted F1 was 0.42. The best result-reported were from the hybrid model *3-parallel Bi-LSTM* with word2vec and random word embedding and random char embeddings where weighted F1 reported was **0.55**. The performance of *3-parallel LSTM,and CNN Bi-LSTM* was reported as 0.53 weighted F1.

For the Malayalam dataset, in conventional models, the performance of Random forest (weighted F1 0.59) was better than SVM (weighted F1 0.56) and Decision tree(weighted F1 0.57). The best performing model was *3-parallel Bi-LSTM* with a random word, random character, and word2vec embeddings with weighted F1 **0.63**. Similarly, For the Kannada dataset, in conventional models, the performance of random forest (weighted F1 0.50) was better than the decision tree (weighted F1 0.47) and SVM (weighted F1 0.46). The best performing model was *3-parallel Bi-LSTM* with word2vec, random word, and random character embeddings with weighted F1 **0.56**.

The best models from the development data evaluation were then submitted into the competition and evaluated by the organizers. On the test data provided by the organizers, they evaluate each model submitted by all participating teams against each task. They provided the data without a label, and the final ranking for all submitted models was published based on our submitted model. The test dataset weighted F1 score produced by the associated model against each task is shown in Table 3. As can be seen in Table Table 3, for Tamil, Malayalam, and Kannada datasets we secured 15th, 12th, and 12th rank among all other submitted models.

4. Conclusion

The current paper identified the sentiment of Dravidian code-mixed YouTube comments written in Tamil, Malayalam, and Kannada. Every YouTube comment was categorized in one of the 5 categories “Positive”, “negative”, “not-tamil/malayalam/kannada”, “unknown_state” and “mixed-feelings”. In all of the three datasets, the best performing model was *3-parallel LSTM* model with Word2vec embedding, random word, and random char embeddings. The model reported weighted F1 0.55,0.63 and 0.56 for development data. The model obtained 15th, 12th, and 12th positions respectively for Tamil, Malayalam, and Kannada datasets.

References

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

[2] D. M. E.-D. M. Hussein, A survey on sentiment analysis challenges, Journal of King Saud University-Engineering Sciences 30 (2018) 330–338.

[3] Q. Zhou, R. Xia, C. Zhang, Online shopping behavior study based on multi-granularity opinion mining: China versus america, Cognitive Computation 8 (2016) 587–602.

[4] S. Saumya, J. P. Singh, Y. K. Dwivedi, Predicting the helpfulness score of online reviews using convolutional neural network, Soft Computing (2019,https://doi.org/10.1007/s00500-019-03851-5) 1–17.

[5] Y. Wang, C. Yu, Social interaction-based consumer decision-making model in social commerce: The role of word of mouth and observational learning, International Journal of Information Management 37 (2017) 179–189.

[6] K. Kenyon-Dean, E. Ahmed, S. Fujimoto, J. Georges-Filteau, C. Glasz, B. Kaur, A. Lalande, S. Bhanderi, R. Belfer, N. Kanagasabai, et al., Sentiment analysis: It's complicated!, in: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), 2018, pp. 1886–1895.

[7] S. Saumya, J. P. Singh, et al., Spam review detection using lstm autoencoder: an unsupervised approach, Electronic Commerce Research (2020) 1–21, https://doi.org/10.1007/s10660–020–09413–4.

[8] K. Ravi, V. Ravi, A survey on opinion mining and sentiment analysis: tasks, approaches and applications, Knowledge-based systems 89 (2015) 14–46.

[9] 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.

[10] S. Banerjee, A. Jayapal, S. Thavareesan, Nuig-shubhanker@dravidian-codemix- fire2020: Sentiment analysis of code-mixed dravidian text using xlnet, in: FIRE, 2020.

[11] 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.

[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] 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.

[14] 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.

[15] 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.

[16] 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.

[17] R. Gatautis, The rise of the platforms: Business model innovation perspectives, Engineering Economics 28 (2017) 585–591.

[18] L. Yue, W. Chen, X. Li, W. Zuo, M. Yin, A survey of sentiment analysis in social media, Knowledge and Information Systems 60 (2019) 617–663.

[19] A. Joshi, A. Prabhu, M. Shrivastava, V. Varma, Towards sub-word level compositions for sentiment analysis of hindi-english code mixed text, in: Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, 2016, pp. 2482–2491.

[20] 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.

[21] B. R. Chakravarthi, N. Jose, S. Suryawanshi, E. Sherly, J. P. McCrae, A sentiment analysis dataset for code-mixed malayalam-english, arXiv preprint arXiv:2006.00210 (2020).