Sentiment Analysis on Code-mixed Dravidian Languages, A Non-linguistic Approach

Prasad A. Joshi[1], Varsha M. Pathak[2]

[1]JET's Zulal Bhilajirao Pail college, Dhule
[2]Institute of Management and Research, Jalgaon

Abstract

Identification of sentiment analysis from social media contents has more attention in the past decades. Such social media contents are code-mixed in nature and peoples find easier to express in this format. They can bind their mother tongue with English. This task deals with identifying sentiment analysis from code-mixed Dravidian languages. Dataset provided by the organisers are in Tamil-English, Kannada-English and Malayalam-English languages. Our system uses the three different approaches viz: machine learning(MNB and DTC), neural-network(ANN and CNN) and transfer learning(BERT, mBERT). For Malayalam-English, MNB, trained using TF-IDF found best. For Tamil-English, ANN and for Kannada-English CNN performed better.

Keywords

sentiment analysis language detection, Code-mixing, MNB, DTC, ANN, CNN, BERT,

1. Introduction

In twentieth century people did not have an easy and comfortable medium for expressing themselves publicly. Twenty first century has begun with a revolutionary shift in information technology with an obvious good or bad impact of its use. In recent years, social media platforms have become an obvious part of lives of a majority of peoples. Popular social media platforms such as YouTube, Facebook, Instagram, Twitter and many more provide a medium of public expressions keeping the required level of privacy. These expressions/comments are usually informal in nature. To such informal comment, many people use blends of two or more languages, which is referred as a code-mixed or code-switched language.

As per the Indian constitution, there are 22 scheduled languages in India. Out of these, Dravidian Languages like Malayalam, Tamil, Kannada have highest speakers and rank within top 10 as per 2011 census data of India. According to the linguists, Malayalam, Tamil and Kannada belong to the South Dravidian language family. Tamil is one of the oldest spoken language of Tamilnadu state of India. Many Tamilian people live in SriLanka and by the Tamil diaspora around the world, whereas Kannada and Malayalam are Dravidian languages spoken in the part of Karnataka, and Kerala [1]. In India, approximately 400 million peoples handling social media out of 1.38 billion, but only 0.02% Indians speak English as their first language. So

FIRE 2021: Forum for Information Retrieval Evaluation, December 13-17, 2021, India
sayprasadajoshi@gmail.com (P. A. Joshi); varsha.pathak@imr.ac.in (V. M. Pathak)
0000-0001-5522-6187 (P. A. Joshi)

most of the peoples in India rely on code-mixed or code-switched languages, which leads to different sentiments such as hate speech [2], insult [3], offensive comments [4] and trolling [5]. If these kind of negative sentiments are not taken care in time, can harm communal health and can turn into devastating events [6]. Specially many code-mixed Indian languages and under-resourced languages [7], needs serious attention. With this motivation, the researchers have initiated their work on identifying different sentiments occurring on social media. In this context, the shared task on sentiment analysis for Dravidian Languages in Code-Mixed Text has been organized [1] and authors have participated. The details about this task and its purpose is given in Section 3.

2. Related Work

From the above discussion we can understand different types of sentiments occurring in code-mixed text content on social media. Though significant amount of sentiment related text in code-mixed Dravidian languages could be found on social media, very less sentiment analysis work has been done till date. Bharathi Raja et al.[8], attracted attention of researchers by organizing special track on Sentiment Analysis for Dravidian Languages in Code-Mixed Text in year 2020. Similarly Bharathi Raja et al. [9], has organized a Shared Task on identifying Offensive Language from Dravidian languages viz.: Tamil, Malayalam, and Kannada in year 2021. In both these task, the organizers have provided training, validation and test datasets with appropriate annotations. In these tasks, many researchers have participated and have applied different machine learning and transfer learning techniques.

Our study shows that prior to the initiatives taken for Dravidian languages, Patra et al., [10] took efforts for code-mixed Hindi-English and code-mixed Bengali-English languages for sentiment analysis. They have organized shared task in ICON 2017 and prepared the dataset using Twitter API. In the year 2018, Aditya Bohra et al. [11], contributed to this research area and they have developed the dataset in code mixed Hindi-English using Twitter Python API. Similarly Anita Saroj and Sukomal Pal [12], have created the dataset in English and code-mixed Hindi languages by using Facebook and Tweeter media in year 2020. This data was collected from parliamentary election of India (PEI-2019) event and they implemented different machine classifiers.

3. About HASOC Shared Task

The goal of this shared task is to categorize the posts/comments of the Dravidian code-mixed dataset collected from YouTube comments into different sentiments polarity. The dataset provided by organizers are Malayalam-English, Tamil-English and Kannada-English languages, containing code-mixed sentences viz. Inter-Sentential switch, Intra-Sentential switch and Tag switching [13]. The Dataset is consisting of 09 tsv files. For each language, the dataset has 03 files for Training, Validation and Testing respectively. Both Training and Validation dataset have three columns namely id, comment/post and sentiment polarity respectively. The polarity for respective comment/post, is annotated with five classes viz. Mixed feeling, Negative, Positive, not-language and unknown state. The test dataset has single text column that contains YouTube

Table 1
Statistics of Dataset

Class	Malayalam			Tamil			Kannada		
	Train.	**Dev.**	**Test**	**Train.**	**Dev.**	**Test**	**Train.**	**Dev.**	**Test**
Mixed_feelings	926	102	Not Known	4020	438	Not Known	574	52	Not Known
Negative	2105	237		4271	480		1188	139	
Positive	6421	706		20070	2257		2823	321	
not-language	1157	141		1667	176		916	110	
unknown_state	5279	580		5628	611		711	69	
Total	15888	1766	1962	35656	3962	4402	6212	691	768

comments/posts. The participants have been asked to develop a system that can identify the appropriate class of respective comment and annotate the test data accordingly [14]. The dataset details, contain total number of comments of five classes. These details of training and development datasets given by the organisers is shown in the Table 1. From this table, we can see that the classes are highly imbalanced.

4. Methodology

As the datasets are collected from social media, they are noisy in nature. So pre-processing is required, which is done at the initial stage of this experimental work. Later features are extracted from datasets(training and validation), using TF-IDF and Keras Tokenizer API [1]. On these extracted features, two different approaches, Machine learning and Neural network, are applied. Results of these two approaches, are submitted to HASOC 2021 shared task. To improve the system further, we have applied a Transfer Learning, of which results were not be submitted to task organiser. Our system is thus based on three approaches as mentioned below. The code is available on GitHub [2].

- Machine learning.
- Neural network.
- Transfer learning.

The performance of all these models is evaluated for **development dataset**, using weighted average F1-score. For all the three Dravidian language, same methodology is followed. The working of each approach is presented in this article in detail.

4.1. Data Pre-processing

We have removed the white spaces, digits, special characters, extra spaces and emojis etc. English stop-words are also removed. For Malayalam language, we have used ml2en algorithm[3]. The ml2en algorithm, transliterates Malayalam script to Roman script ('Manglish').

[1]https://keras.io/
[2]https://github.com/sayprasad1/KBCNMUJAL-HASOC-2021
[3]https://nadh.in/code/ml2en/

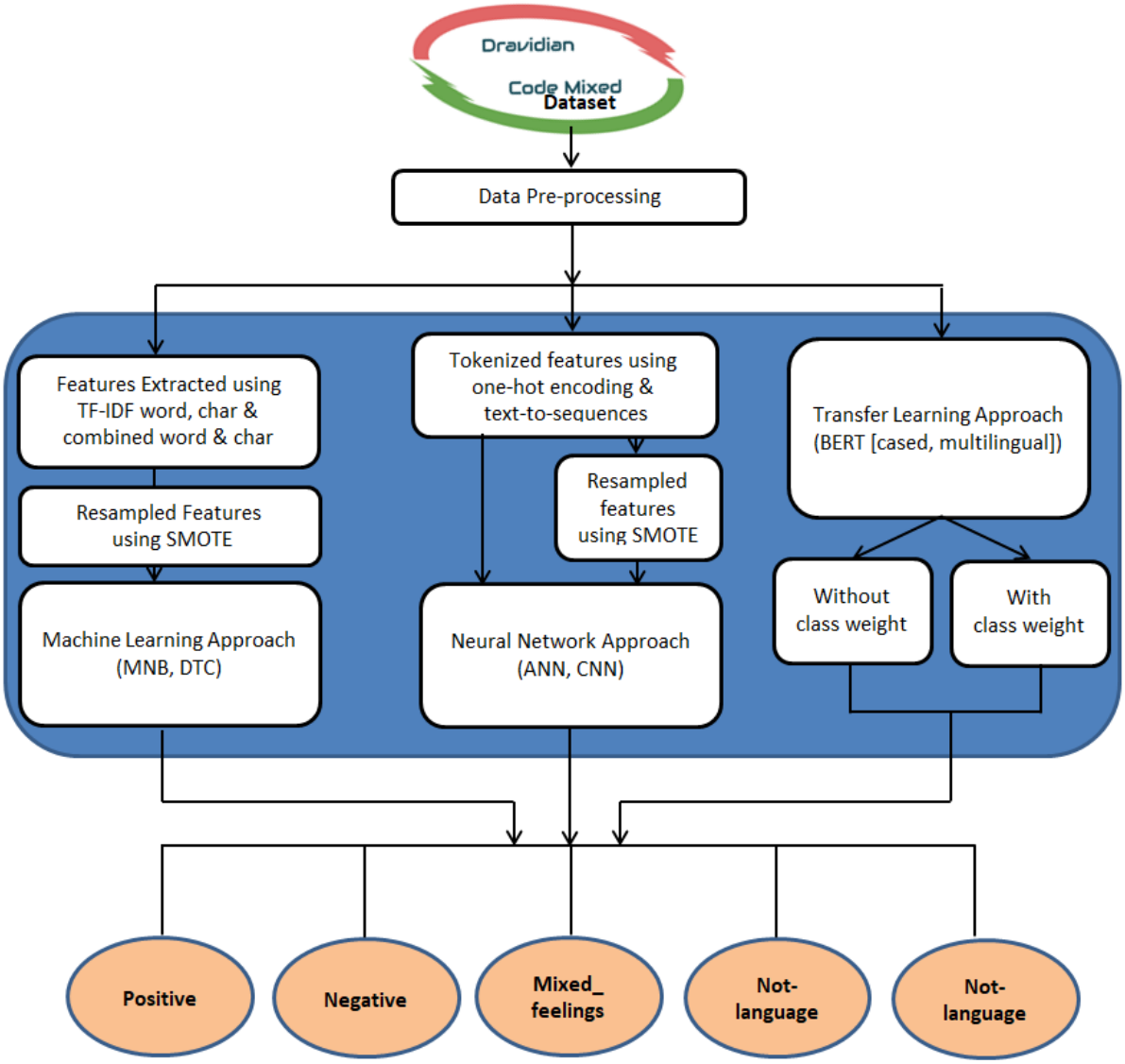

Figure 1: HASOC-2021:kbcnmujal system: An architectural view

4.2. Machine learning based approach

We have applied different n-gram ranges of TF-IDF word, TF-IDF character and TF-IDF combined word and character for extracting the features. The selection of different ranges is done, by testing at what range the classifier is producing the highest f1-score. For every language, we have tested the different n-gram range, and finally, have used the following n-gram range .

- For Malayalam word n-gram of (1,1) and character n-gram range of (5,5) were applied.

Table 2
Results for the Machine learning approach with different word n-gram range of TF-IDF feature

Classifier	Class	Malayalam n-gram range (1,1)			Tamil n-gram range (5,6)			Kannada n-gram range (4,5)		
		Preci.	Rec.	F1	Preci.	Rec.	F1	Preci.	Rec.	F1
MNB	Mixed_feelings	0.70	0.64	0.67	0.20	0.95	0.33	0.20	0.97	0.33
	Negative	0.70	0.72	0.71	0.47	0.01	0.02	0.73	0.02	0.05
	Positive	0.65	0.75	0.70	0.45	0.06	0.11	0.58	0.04	0.08
	not-language	0.89	0.93	0.91	0.96	0.05	0.09	1.00	0.02	0.05
	unknown_state	0.70	0.60	0.95	0.58	0.05	0.08	0.00	0.00	0.00
	Weighted avg	0.73	0.73	0.73	0.53	0.22	0.13	0.50	0.21	0.10
DTC	Mixed_feelings	0.66	0.31	0.42	0.44	0.01	0.02	0.92	0.03	0.07
	Negative	0.61	0.47	0.53	0.09	0.00	0.00	0.67	0.02	0.04
	Positive	0.39	0.64	0.49	0.65	0.04	0.08	0.70	0.02	0.04
	not-language	0.71	0.75	0.73	0.20	0.99	0.34	0.20	1.00	0.34
	unknown_state	0.49	0.53	0.51	0.69	0.03	0.06	0.00	0.00	0.00
	Weighted avg	0.57	0.54	0.54	0.42	0.21	0.10	0.50	0.21	0.10

- For Tamil word n-gram of (5,6) and character n-gram range of (5,6) were applied.
- For Kannada word n-gram of (4,5) and character n-gram range of (4,5) were applied.

After extracting features using above TF-IDF n-gram ranges, Multinomial Naive Bayes (MNB) and Decision Tree Classifiers (DTC) are implemented. For MNB we have set 'alpha' parameter in 0.5 to 2.0 range. For DTC, we have set the 'criterion' parameter with 'gini' and 'entropy' values. The rest of the hyper parameters of both these classifiers are kept at their default values.

As the classes have severe skew in distribution, both the classifiers failed to categoriz the comments among the five classes for all the Dravidian languages. Hence SMOTE[4] technique, is applied on TF-IDF extracted features. SMOTE is the most popular oversampling method. While applying SMOTE on TF-IDF extracted features sampling strategy is kept to auto and thus we get the re-sampled TF-IDF features according to the majority class. On re-sampled features MNB and DTC were applied. While applying DTC for Tamil language we have selected 10,000 features because training and development dataset have bigger size as compared to Malayalam and Kannada. The class-wise performance of both the classifiers on this re-sampled features, is given in the Table 2, 3, 4.

4.3. Neural network based approach

We have implemented an Artificial Neural Network (ANN) and Convolution Neural Network (CNN) in our work. For ANN we created tokenization of datasets using one-hot encoded matrix. For one-hot encoded matrix, we have used keras text_to_matrix method. For CNN, we have used keras text_to_sequences method for the tokenization. The extracted tokens are used to construct the vocabulary base of the respective language.

[4]imblearn.over_sampling.SMOTE

Table 3
Results for the Machine learning approach with different character n-gram range of TF-IDF feature

Classifier	Class	Malayalam n-gram range (5,5)			Tamil n-gram range (5,6)			Kannada n-gram range (4,5)		
		Preci.	Rec.	F1	Preci.	Rec.	F1	Preci.	Rec.	F1
MNB	Mixed_feelings	0.77	0.73	0.75	0.40	0.38	0.39	0.51	0.21	0.30
	Negative	0.70	0.76	0.73	0.51	0.62	0.56	0.48	0.87	0.62
	Positive	0.66	0.79	0.72	0.51	0.59	0.54	0.53	0.61	0.57
	not-language	0.93	0.89	0.91	0.84	0.80	0.82	0.66	0.82	0.73
	unknown_state	0.74	0.60	0.66	0.61	0.44	0.51	0.72	0.28	0.40
	Weighted avg	0.76	0.75	0.75	0.57	0.57	0.56	0.58	0.56	0.52
DTC	Mixed_feelings	0.40	0.31	0.35	0.37	0.30	0.33	0.40	0.30	0.34
	Negative	0.35	0.34	0.35	0.40	0.38	0.39	0.45	0.56	0.50
	Positive	0.31	0.41	0.36	0.44	0.63	0.52	0.43	0.56	0.49
	not-language	0.65	0.49	0.56	0.72	0.59	0.65	0.64	0.64	0.64
	unknown_state	0.31	0.37	0.34	0.38	0.39	0.38	0.57	0.40	0.47
	Weighted avg	0.41	0.38	0.39	0.46	0.45	0.45	0.50	0.49	0.49

Table 4
Results for the Machine learning approach with combined word and character n-gram range of TF-IDF feature

Classifier	Class	Malayalam			Tamil			Kannada		
		Preci.	**Rec.**	**F1**	**Preci.**	**Rec.**	**F1**	**Preci.**	**Rec.**	**F1**
MNB	Mixed_feelings	0.79	0.75	0.77	0.42	0.30	0.35	0.48	0.22	0.31
	Negative	0.75	0.75	0.75	0.53	0.58	0.55	0.56	0.84	0.67
	Positive	0.69	0.80	0.74	0.45	0.68	0.54	0.47	0.66	0.55
	not-language	0.93	0.92	0.93	0.85	075	0.80	0.68	0.83	0.75
	unknown_state	0.74	0.66	0.70	0.55	0.44	0.49	0.74	0.31	0.43
	Weighted avg	**0.78**	**0.78**	**0.78**	0.56	0.55	0.55	0.59	0.57	0.54
DTC	Mixed_feelings	0.56	0.39	0.46	0.37	0.31	0.34	0.30	0.22	0.26
	Negative	0.48	0.52	0.50	0.40	0.35	0.38	0.48	0.53	0.51
	Positive	0.40	0.52	0.45	0.43	0.63	0.51	0.43	0.63	0.51
	not-language	0.78	0.65	0.71	0.72	0.58	0.64	0.57	0.58	0.57
	unknown_state	0.46	0.50	0.48	0.38	0.38	0.38	0.61	0.40	0.49
	Weighted avg	0.54	0.52	0.52	0.46	0.45	0.45	0.48	0.47	0.47

After tokenizing the text, ANN is built by using three dense layers. The first input (dense) layer has 512 nodes and input vocabulary of size 10,000 is provided. This is followed by activation layer using relu activation function, which is followed by dropout layer of 0.3. The second dense layer also has the 512 node followed by the activation layer having relu function. This is followed by a dropout layer with dropout of 0.3. The last output (dense) layer has 5 nodes because we have to be categorized the comments/post into five classes. Dense layer is followed by activation layer using sigmoid as activation function. The proposed ANN trained with categorical_crossentropy loss function and Adam optimizer. The training has the batch size of 32 with 10 epochs. The same ANN network is implemented for all the three Dravidian

Table 5
Results of the ANN and CNN approach

Classifier	Class	Malayalam			Tamil			Kannada		
		Preci.	Rec.	F1	Preci.	Rec.	F1	Preci.	Rec.	F1
ANN	Mixed_feelings	0.51	0.40	0.45	0.21	0.17	0.19	0.22	0.19	0.21
	Negative	0.54	0.55	0.55	0.39	0.36	0.38	0.59	0.55	0.57
	Positive	0.76	0.76	0.76	0.72	0.78	0.75	0.69	0.74	0.72
	not-language	0.67	0.69	0.68	0.55	0.51	0.53	0.62	0.59	0.60
	unknown_state	0.68	0.69	0.68	0.41	0.36	0.38	0.49	0.46	0.48
	Weighted avg	0.68	0.68	0.68	**0.57**	**0.59**	**0.58**	0.60	0.61	0.61
CNN	Mixed_feelings	0.53	0.31	0.40	0.22	0.13	0.16	0.62	0.10	0.17
	Negative	0.69	0.51	0.59	0.44	0.32	0.37	0.64	0.56	0.60
	Positive	0.72	0.83	0.77	0.70	0.82	0.75	0.67	0.83	0.74
	not-language	0.73	0.76	0.75	0.67	0.53	0.59	0.60	0.62	0.61
	unknown_state	0.72	0.71	0.71	0.38	0.35	0.37	0.46	0.32	0.38
	Weighted avg	0.71	0.71	0.70	0.56	0.60	0.57	**0.63**	**0.64**	**0.61**

Table 6
Results of the ANN and CNN approach after SMOTE

Classifier	Class	Malayalam			Tamil			Kannada		
		Preci.	Rec.	F1	Preci.	Rec.	F1	Preci.	Rec.	F1
ANN	Mixed_feelings	0.81	0.44	0.57	0.24	0.18	0.21	0.67	0.50	0.58
	Negative	0.44	0.53	0.48	0.38	0.37	0.38	0.56	0.58	0.57
	Positive	0.61	0.80	0.69	0.62	0.80	0.70	0.58	0.66	0.62
	not-language	0.75	0.64	0.69	0.95	0.74	0.84	0.50	0.62	0.55
	unknown_state	0.58	0.65	0.61	0.37	0.35	0.36	0.33	0.33	0.33
	Weighted avg	0.65	0.62	0.62	*0.67*	*0.65*	*0.65*	0.58	0.57	0.57
CNN	Mixed_feelings	0.53	0.59	0.56	0.18	0.11	0.14	0.51	0.59	0.55
	Negative	0.58	0.44	0.50	0.40	0.28	0.33	0.50	0.27	0.35
	Positive	0.64	0.65	0.65	0.59	0.65	0.62	0.57	0.60	0.58
	not-language	0.73	0.70	0.71	0.63	0.70	0.60	0.56	0.57	0.57
	unknown_state	0.61	0.58	0.60	0.36	0.26	0.30	0.46	0.41	0.43
	Weighted avg	0.60	0.60	0.60	0.54	0.56	0.54	0.53	0.53	0.52

languages.

For implementing CNN, we have applied different parameters for all the Dravidian languages. We have extracted total unique words(vocab_size) and length of longest sentence(max_length) from the dataset for all languages. We found 40230, 69675 and 15800 unique words from Malayalam, Tamil and Kannada languages respectively. For Malayalam, Tamil and Kannada maximum length of comment/post is 195, 124 and 92 respectively. Every input comment padded with a maximum length(max_length) of that language. For every language, embedding of 100 is used with unique words(vocab_size). So for Malayalam, Tamil and Kannada, we get (40230 X 100), (69675 X 100) and (15800 X 100) dimensional embedding matrix respectively. Thus embedding layer is different for each language, but the next set of layers(Conv1D, GlobalMaxPooling1D and dense) are same. Embedding layer treated as input to Conv1D layer. Conv1D provided with

Table 7
Results of the Transfer Learning

Classifier	Class	Malayalam			Tamil			Kannada		
		Preci.	Rec.	F1	Preci.	Rec.	F1	Preci.	Rec.	F1
BERT	Mixed_feelings	0.50	0.38	0.43	0.28	0.21	0.24	0.22	0.21	0.21
	Negative	0.61	0.57	0.59	0.42	0.36	0.39	0.73	0.53	0.62
	Positive	0.76	0.82	0.79	0.72	0.82	0.77	0.70	0.80	0.75
	not-language	0.87	0.84	0.85	0.56	0.52	0.54	0.65	0.66	0.66
	unknown_state	0.74	0.72	0.73	0.47	0.37	0.41	0.52	0.46	0.49
	Weighted avg	0.73	0.73	0.73	0.59	0.62	0.60	*0.65*	*0.64*	*0.64*
mBERT	Mixed_feelings	0.44	0.39	0.42	0.48	0.14	0.21	0.29	0.19	0.23
	Negative	0.68	0.61	0.64	0.45	0.34	0.39	0.62	0.68	0.65
	Positive	0.79	0.79	0.79	0.66	0.91	0.77	0.73	0.73	0.73
	not-language	0.85	0.84	0.84	0.70	0.44	0.54	0.65	0.70	0.67
	unknown_state	0.73	0.78	0.75	0.52	0.23	0.32	0.46	0.43	0.45
	Weighted avg	*0.74*	*0.74*	*0.74*	*0.60*	*0.63*	*0.61*	0.64	0.65	0.64

Table 8
Results of the Transfer Learning using Class Weights

Classifier	Class	Malayalam			Tamil			Kannada		
		Preci.	Rec.	F1	Preci.	Rec.	F1	Preci.	Rec.	F1
BERT	Mixed_feelings	0.44	0.40	0.42	0.26	0.35	0.30	0.21	0.27	0.24
	Negative	0.62	0.60	0.61	0.38	0.36	0.37	0.66	0.52	0.62
	Positive	0.79	0.79	0.79	0.76	0.69	0.73	0.76	0.75	0.75
	not-language	0.85	0.83	0.84	0.49	0.54	0.51	0.65	0.72	0.68
	unknown_state	0.74	0.76	0.75	0.40	0.44	0.42	0.56	0.51	0.53
	Weighted avg	0.73	0.73	0.73	0.59	0.57	0.58	0.66	0.65	0.65
mBERT	Mixed_feelings	0.42	0.46	0.44	0.24	0.36	0.29	0.14	0.21	0.17
	Negative	0.60	0.56	0.52	0.41	0.42	0.42	0.62	0.61	0.61
	Positive	0.79	0.78	0.78	0.80	0.67	0.73	0.81	0.72	0.76
	not-language	0.79	0.87	0.83	0.54	0.60	0.56	0.68	0.79	0.73
	unknown_state	0.73	0.72	0.73	0.39	0.47	0.43	0.52	0.49	0.50
	Weighted avg	0.72	0.72	0.72	0.62	0.57	0.59	*0.67*	*0.65*	*0.66*

number of output filters as 64, and kernel size of 3 with relu as activation function. Conv1D layer is used for generating sequences of text. Conv1D layer is followed by GlobalMaxPooling1D layer and dropout of 0.5. The dense layer has 5 nodes with activation function as softmax. CNN trained with categorical_crossentropy loss function and Adam optimizer. The training has the batch size of 128 with 5 epochs.

Though the classes were highly imbalanced in nature, neural-network approach(ANN and CNN) successfully categorized the post into the five classes for all the Dravidian languages. The class wise distribution of the dataset using ANN and CNN is give in Table 5.

Further to analyse the performance of ANN and CNN on resampled dataset, We have applied SMOTE technique on the tokenized text(features). While applying SMOTE, sampling strategy parameter is set to minority, which resamples the minority class. Thus we have received the

resampled features according to the minority class. On resampled features, the same ANN and CNN are implemented. This has enabled to predict the comments among the desired classes. The results are presented in Table 6.

After comparing the result of neural network approaches on tokenized features and on resampled features from Table 5 and Table 6, we can observe that for Malayalam and Kannada language results are very low for resampled features, but for Tamil language ANN performed better on resampled features.

4.4. Transfer learning

For applying transfer learning, we used Simple Transformers [15] library based on the Transformers library by HuggingFace. Simple Transformers permits to fine-tune Transformer models. Transformers offers plenty of pretrained models which are used to accomplish different tasks like text classification, multi-label text classification, text generation, question answering, summarization, translation, information extraction and more in over 100 languages. Our approach used two different variations of BERT [16] (Bidirectional Encoder Representations from Transformers) transfer models, for categorizing the comments. The two variations are :

- Pretrained BERT base model (bert-base-cased) : 12-layer, 768-hidden, 12-heads, 110M parameters. It is trained on lower-cased English text using a masked modelling technique.
- Pretrained BERT multilingual model (bert-base-multilingual-cased) : It has 12 layers, 768 hidden, 12 attention heads with 168M parameters. It is trained on lower-cased text in 104 languages with masked language modeling.

For implementing above models we created instance of ClassificationModel with its parameters and their values. For each language we created two different models : model handling class imbalance while other model does not handle class imbalance. The model handling class imbalance has weight attribute, which takes input as list of weight for each label. To calculate weights of the classes we implemented compute_class_weight[5] from sklearn. For both the models hyperparameter 'fp16' is set to 'false' while rest of the parameters are set to their default values. Both these models are trained by using training dataset with 4 epochs and results were evaluated using development dataset. Table 7 shows the results of BERT models not handling class imbalance while BERT models handling class imbalance are shown in Table 8.

After examining Table 7 and Table 8, we have noticed that, class weights have dropped the performance for Tamil language, whereas for Kannada language, the results show that there is slight improvement for both BERT models. For Malayalam language, both BERT models with class weights and without class weights have produced same result, while BERT-multilingual without class weights have shown better performance than BERT-multilingual with class weights.

5. Result and discussion

The system performance is evaluated in terms of precision, recall, and F1-score for all the five sentiment classes. The weighted average of these classes is also given. Among the three

[5] sklearn.utils.class_weight.compute_class_weight

approaches, the one, which has highest weighted average of precision, recall and F1-score, is considered as best. The results of best performing model were submitted to the task organiser. As mentioned earlier, results of transfer learning and neural network approach using resampled features were not submitted.

For Malayalam language, among all the classifiers, MNB using combined word n-gram range(1,1) and character n-gram range(5,5) performed with better results and got the precision, recall and F1-score of 0.78. MNB with word n-gram range(1,1) and MNB with character n-gram range(5,5) also performed well with F1-score of 0.73 and 0.75 respectively. In the same machine learning approach, results of DTC are very poor, with word n-gram and combined word and character n-gram it achieves F1-score of 0.54 and 0.52 respectively. But DTC with character n-gram range(5,5) not even scored 50% weighted average of precision, recall and F1-score. The Neural network approach performed very well using tokenized features as compared to resampled features because weighted average F1-score dropped from 0.68 to 0.62 for ANN and for CNN it dropped from 0.70 to 0.60. Both BERT-models with and without class weights have same weighted average precision, recall and F1-score of 0.73, whereas BERT-multilingual without class weights have weighted average F1-score of 0.74 and 0.73 with class weights.

In case of Tamil language, ANN using resampled features performed better with F1 score of a 0.65 as compared to tokenized features with F1 score 0.58. CNN using tokenized features have F1 score 0.57 and using resampled features it shows a degradation of 0.54. MNB using character n-gram range(5,6) and using combined word n-gram range(5,6) and character n-gram(5,6) have near about same F1 score of 0.55. Performance of MNB and DTC with word n-gram range(5,6) were very low with F1-score of 0.13 and 0.10 respectively. DTC with character n-gram range(5,6) and DTC with combined word n-gram range(5,6) and character n-gram range(5,6) has not shown promising results with F1-score of 0.45. Both BERT models without class weights performed marginally better. BERT-multilingual without class weight have overshadow the performance of BERT-based with higher precision, recall and F1-score of 0.60, 0.63 and 0.61 respectively.

CNN and ANN have shown better results using tokenized features as compared to resampled features. CNN using tokenized features has weighted average precision and recall values as 0.63 and 0.64 respectively in case of Kannada language. Similarly, for Tamil language, MNB and DTC with word n-gram range(4,5) has received very low F1-score i.e. 0.10. DTC with character n-gram and DTC with combined word n-gram and character n-gram have shown good F1-score as compared to Malayalam and Tamil language. BERT-based and BERT-multilingual models without class weights have shown equal results with F1-score as 0.64. Both the transfer learning models have performed better, using class weights. BERT-multilingual model has recei better weighted average F1 score i.e. 0.66.

From all the above observations, we can conclude that, in case of Malayalam language, machine learning model: MNB using combined word n-gram and character n-gram has scored the highest results. For Tamil language, the neural network model: ANN using resampled features has scored the better results. In case of Kannada language BERT-multilingual using class weights performed better.

6. Conclusion

From these results, we can report that Roman/Latin script helps in improving the system performance for all the three approaches. Machine learning approach with word n-gram feature categorized the sentiments, only for Roman/Lain script(e.g, Manglish). On the other hand, for code-mixed sentences(e.g.Tanglish and Tamil), it failed miserably. On the contrary, the same machine learning approach with character n-gram feature and combined word and character n-gram feature successfully categorized the sentiments irrespective whether it is Roman/Lain script or code-mixed sentences. Experiment of the system shows that class imbalanced is successfully handled by neural-network and transfer learning, whereas for machine learning approach resampling of features is must.

References

[1] 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.

[2] T. Davidson, D. Warmsley, M. W. Macy, I. Weber, Automated hate speech detection and the problem of offensive language, CoRR abs/1703.04009 (2017). URL: http://arxiv.org/abs/1703.04009. arXiv:1703.04009.

[3] Z. Waseem, Are you a racist or am I seeing things? annotator influence on hate speech detection on Twitter, in: Proceedings of the First Workshop on NLP and Computational Social Science, Association for Computational Linguistics, Austin, Texas, 2016, pp. 138–142. URL: https://aclanthology.org/W16-5618. doi:10.18653/v1/W16-5618.

[4] M. Zampieri, S. Malmasi, P. Nakov, S. Rosenthal, N. Farra, R. Kumar, SemEval-2019 task 6: Identifying and categorizing offensive language in social media (OffensEval), in: Proceedings of the 13th International Workshop on Semantic Evaluation, Association for Computational Linguistics, Minneapolis, Minnesota, USA, 2019, pp. 75–86. URL: https://aclanthology.org/S19-2010. doi:10.18653/v1/S19-2010.

[5] A. Koufakou, V. Basile, V. Patti, FlorUniTo@TRAC-2: Retrofitting word embeddings on an abusive lexicon for aggressive language detection, in: Proceedings of the Second Workshop on Trolling, Aggression and Cyberbullying, European Language Resources Association (ELRA), Marseille, France, 2020, pp. 106–112. URL: https://aclanthology.org/2020.trac-1.17.

[6] V. M. Pathak, M. Joshi, P. Joshi, M. Mundada, T. Joshi, Kbcnmujal@hasoc-dravidian-codemix-fire2020: Using machine learning for detection of hate speech and offensive code-mixed social media text, CoRR abs/2102.09866 (2021). URL: https://arxiv.org/abs/2102.09866. arXiv:2102.09866.

[7] 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.

[8] 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.

[9] 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.

[10] B. G. Patra, D. Das, A. Das, Sentiment analysis of code-mixed indian languages: An overview of sail_code-mixed shared task @icon-2017, ArXiv abs/1803.06745 (2018).

[11] A. Bohra, D. Vijay, V. Singh, S. S. Akhtar, M. Shrivastava, A dataset of Hindi-English code-mixed social media text for hate speech detection, in: Proceedings of the Second Workshop on Computational Modeling of People's Opinions, Personality, and Emotions in Social Media, Association for Computational Linguistics, New Orleans, Louisiana, USA, 2018, pp. 36–41. URL: https://aclanthology.org/W18-1105. doi:10.18653/v1/W18-1105.

[12] A. Saroj, S. Pal, An Indian language social media collection for hate and offensive speech, in: Proceedings of the Workshop on Resources and Techniques for User and Author Profiling in Abusive Language, European Language Resources Association (ELRA), Marseille, France, 2020, pp. 2–8. URL: https://aclanthology.org/2020.restup-1.2.

[13] 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.

[14] B. R. Chakravarthi, R. Priyadharshini, S. Thavareesan, D. Chinnappa, D. Thenmozhi, E. Sherly, J. P. McCrae, A. Hande, R. Ponnusamy, S. Banerjee, C. Vasantharajan, Findings of the Sentiment Analysis of Dravidian Languages in Code-Mixed Text, in: Working Notes of FIRE 2021 - Forum for Information Retrieval Evaluation, CEUR, 2021.

[15] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, A. M. Rush, Transformers: State-of-the-art natural language processing, in: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, Association for Computational Linguistics, Online, 2020, pp. 38–45. URL: https://www.aclweb.org/anthology/2020.emnlp-demos.6.

[16] J. Devlin, M. Chang, K. Lee, K. Toutanova, BERT: pre-training of deep bidirectional transformers for language understanding, CoRR abs/1810.04805 (2018). URL: http://arxiv.org/abs/1810.04805. arXiv:1810.04805.