CENMates@HASOC-Dravidian-CodeMix-FIRE2020: Offensive Language Identification on Code-mixed Social Media Comments

Veena P. V.[a], Praveena Ramanan[b] and Remmiya Devi G [c]

[a]*Freelancer*
[b]*Freelancer*
[c]*Freelancer*

Abstract

This paper presents the working methodology and results on offensive language identification on Dravidian code-mixed data for the shared task of FIRE 2020. This task aims at identifying whether the comments written on social media platforms are offensive or not. The shared task contains Malayalam code-mixed comments, and Manglish (Malayalam written in Roman script) and Tanglish (Tamil written in Roman script) comments. Identification of hate speech on social media data has become an interesting domain of research and hence there are several ongoing researches happening for the same. The dataset for the HASOC task 1 has been retrieved from YouTube and for task 2 from YouTube and Twitter. TF-IDF vectors along with character level n-grams are passed as features to the proposed system for system development. We developed and evaluated four systems consisting of Logistic regression, XGBoost, Long Short Term Memory networks, and Attention networks. Amongst the tasks performed, the best results were obtained with an F1 score of 0.93 for Task 2 Malayalam.

Keywords

Offensive Language Identification, Code-mix, Manglish, Tanglish, Machine Learning, Attention network, LSTM, Hate and Speech Offense

1. Introduction

Offensive language implies to any message that conveys with a tone of insult, hatred, rude or anger. This activity should not be encouraged especially if it is happening in a platform that is opened to the public. In legal terms also, there is a high need to address this problem of people using inappropriate words in public to show negative emotions against other people. This paper presents the working notes for Hate Speech and Offensive Content identification in Dravidian languages (HASOC) [1, 2]. This is a work organized by Forum for Information Retrieval and Evaluation (FIRE) 2020. There are two main tasks involved in this work. Task 1 is to identify offensive languages in Malayalam and Task 2 has two subtasks. The first and second subtask in Task 2 is to identify offensive speech contents in Malayalam and Tamil written in Roman characters respectively.

The main idea behind the proposed system is to classify the sentence in the dataset with a label as offensive or non-offensive, by examining the presence of offensive content in the sentence. Although there are researches going on for identifying offensive or hate speech content in English and many other foreign languages, FIRE 2020 has brought up an opportunity to figure out the possibility of solving the same on Dravidian code-mixed languages like Malayalam and Tamil.

In Section 2 of the paper, we mention few related research papers for offensive identification and code-mixed languages. The description of the task and dataset is described in Section 3. Section 4 briefly explains the features and the models we used for developing our systems. Finally, in Section 5, we demonstrate the results and conclusions derived.

2. Related Works

The constant increase in the amount of offensive language in social media has made NLP researchers to propose new models to identify the hate and offensive contents in them. For the Greek language, an offensive language identification was implemented with several computational models [3]. Offensive language identification using a large-scale semi-supervised dataset was experimented with SOLID and OLID datasets yielding improved performance in [4]. In [5] a system was proposed that utilized Deep learning models like LSTM for categorizing offensive language in social media. Ahn et al. proposed a new metric known as Translation Embedding Distance along with methods to improvise pre-trained multilingual BERT (mBERT) in offensive language identification and obtained significant performance [6]. Support Vector Machine (SVM) – Radial Basis Function (RBF) based classifier model was proposed that detected hate speech in Hindi-English code mix data extracted from Facebook's pre-trained word embedding library in [7]. Different machine learning models were experimented with annotated tweets of the offensive language identification dataset (OLID) [8]. Identification of offensive language for English, German, and Hindi code-mixed data using deep learning models that relies on word and context embedding was developed [9]. Other than the languages used by wide users of social media like English, a lot of research has also been going on for the development of social media data of the under-resourced languages [10, 11, 12].

3. Task and Dataset Description

The shared task presents a new gold standard corpus for offensive language detection of code-mixed text in Dravidian languages (Malayalam-English and Tamil-English). Out of the two tasks, Task 1 is intended to classify the YouTube comment into offensive or non-offensive. Task 2 deals with classifying a tweet or YouTube comment in Tanglish and Manglish (Tamil and Malayalam written using Roman Characters).

The organizers provided training, development, and testing datasets separately for Task 1. The training and development data consisted of Comment and the Label fields whereas Test data had ID and Comment fields. Task 2 contains two datasets each for training and testing for Malayalam-English (Manglish) code-mixed data and Tamil-English (Tanglish) code-mixed data.

Table 1
Dataset Statistics

Task	Data	Total	Not Offensive	Offensive
Task-1	Train Data	3200	2633	567
(Malayalam-mix)	Dev Data	400	328	72
Task 2(Manglish)	Train Data	4000	2047	1953
Task 2 (Tanglish)	Train Data	4000	2020	1980

Both Manglish and Tanglish data contain the ID, Tweets, and Label fields. The data statistics of each task is tabulated in Table 1.

Test data of Malayalam mixed data consisted of 1000 comments. Manglish consisted of 1000 whereas Tanglish consisted of 940 comments.

4. System Description

This section briefly explains the systems developed for our submissions in the task. The data was noisy and hence pre-processing had to be performed before training and testing the model. Below are some of the pre-processing steps performed to clean the dataset.

1. Tokenization
2. Lowercase conversion
3. Removal of punctuation and special characters
4. Removal of emojis

4.1. Feature Description

For developing the classification system, one of the main features used was Term Frequency-Inverse Document Frequency (TF-IDF). Term frequency is about the number of times a word has occurred in a document. This can be calculated by having a count of the word appeared in the document. Inverse document frequency is locating the documents that have the word. If the value is too close to 0 that shows the word is more common.

We used TDIDFVectorizer() function from sklearn to fit the data. Character-level with n-grams in the range 3 to 5 were considered as features. The maximum features were limited to 8000 and were weighted with TF-IDF values.

4.2. System Implementation

Four systems were implemented in total for the two tasks. In the following subsections we describe each one of them system briefly.

4.2.1. System 1: Logistic Regression

Logistic Regression (LR) is an algorithm used for classification. The aim of LR is to predict the probability of the event occurring by fitting data to a logit function or a sigmoid. The output

will be a probabilistic value between 0 and 1.

Each cleaned sentence was taken as input and transformed into its corresponding TF-IDF vectors. As a part of feature enrichment, character n-grams were added and given to the LR classifier. Since the data was highly imbalanced, we tried to balance the class weights. LR with character-level features was used as one of the systems for all the three tasks since it produced relatively good results.

4.2.2. System 2: XGBoost

Our second system was implemented using XGBoost. XGBoost known as eXtreme Gradient Boosting is an ensemble learning methodology. It combines the prediction ability of several machine learning models and demonstrated better performance on the validation data. As features for this algorithm, the same TF-IDF vectors explained before was used. This system was used as one of the submissions for both of the task 2 data.

4.2.3. System 3: Long Short-Term Memory (LSTM)

LSTMs are a special type of Recurrent Neural Networks (RNN) capable of handling long-term dependencies which are very important in the case of text data [13]. We used Keras Sequential API for implementing LSTM with TensorFlow as the backend. We used a simple LSTM network with 4 layers which is illustrated in Figure 1.
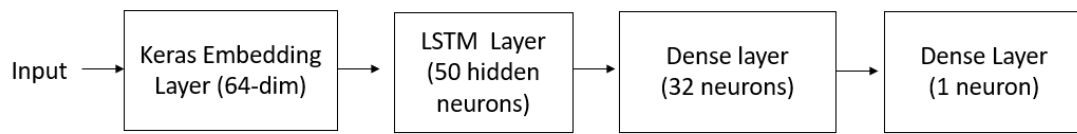
**Figure 1:** Simple LSTM network used for System 3

Each word in the input data is first converted to a one-hot encoded vector. The sentences are then passed through the Embedding layer followed by a LSTM layer. The first dense layer consisted of 32 neurons with ReLU activation and the second dense layer consisted of 1 neuron with Sigmoid activation function.

The model was trained for 10 epochs. Since the output was binary, binary cross-entropy was used as the loss function with Adam as the optimizer.

4.2.4. System 4: Attention with LSTM

Attention networks with respect to standard LSTM network have an advantage that selective attention can be provided to those inputs which create a greater impact. Yang et al. proposed a hierarchical attention-based network which has the ability to provide attention by considering the word both at word level and sentence level [14]. Our work for this system was inspired from this paper aiming to give more attention to the offensive words in each comment. We used a 6-layer network for the implementation of this system which is shown in Figure 2.

1. Embedding Layer with embedding vector size of 64
2. Bidirectional LSTM Layer with number of hidden neurons as 64
3. Bidirectional LSTM Layer with number of hidden neurons as 32
4. Hierarchical Attention Layer
5. A Dense layer of 16 neurons with ReLU activation
6. A Dense Layer of 1 neuron with Sigmoid activation
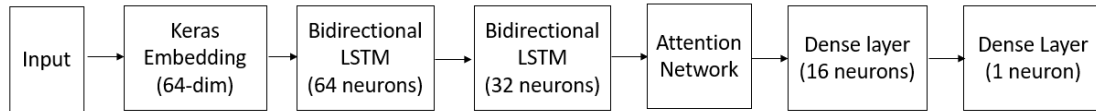
Figure 2: System 4: Attention with LSTM

The model was trained for 10 epochs with Adam as optimizer and binary cross-entropy as the loss function.

5. Results & Conclusion

The organizers used a weighted F1 score for the evaluation of test submissions. The results obtained by the top 3 teams for task 1 for classification of Malayalam code-mixed data is tabulated in Table 2 whereas Table 3 and Table 4 shows the result of the top 3 and top 4 ranks for task 2 Manglish and Tanglish respectively.

Table 2
Final result of Top-3 teams for task 1

Team Name	Precision	Recall	F1- score	Rank
SivaSai@BITS	0.95	0.95	0.95	1
IIITG-ADBU	0.95	0.95	0.95	1
CFILT_IITBOMB	0.94	0.94	0.94	2
SSNCSE-NLP	0.94	0.94	0.94	2
CENMates	**0.93**	**0.93**	**0.93**	**3**
NIT-AI-NLP	0.93	0.93	0.93	3
YUN	0.93	0.93	0.93	3
Zyy1510	0.93	0.93	0.93	3

Our team got first, third and fourth positions in the three tasks. Since the data we used for training was less, We could also see that a simple TD-IDF approach with character level n-gram features using machine learning classifiers was producing good results almost the same as the results obtained with deep learning-based classifiers.

As future work, using additional data, the deep learning classifiers can be evaluated. Different embedding approaches can also be used and evaluated to increase the model performance.

Table 3
Final result of Top-3 teams for task 2 Manglish

Team Name	Precision	Recall	F1- score	Rank
CENMates	**0.78**	**0.78**	**0.78**	**1**
SivaSai@BITS	0.79	0.75	0.77	2
KBCNMUJAL	0.77	0.77	0.77	2
IIITG-ADBU	0.77	0.76	0.76	3

Table 4
Final result of Top-4 teams for task 2 Tanglish

Team Name	Precision	Recall	F1- score	Rank
SivaSai@BITS	0.90	0.90	0.90	1
SSNCSE-NLP	0.88	0.88	0.88	2
Gauravarora	0.88	0.88	0.88	2
KBCNMUJAL	0.87	0.87	0.87	3
IITG-ADBU	0.87	0.87	0.87	3
zyy1510	0.88	0.87	0.87	3
CENMates	**0.86**	**0.86**	**0.86**	**4**

6. Acknowledgments

Acknowledgments

References

[1] B. R. Chakravarthi, M. A. Kumar, J. P. McCrae, P. B, S. KP, T. Mandl, Overview of the track on Hasoc-Offensive Language Identification- DravidianCodeMix, in: Working Notes of the Forum for Information Retrieval Evaluation (FIRE 2020). CEUR Workshop Proceedings. In: CEUR-WS. org, Hyderabad, India, 2020.

[2] B. R. Chakravarthi, M. A. Kumar, J. P. McCrae, P. B, S. KP, T. Mandl, Overview of the track on Hasoc-Offensive Language Identification- DravidianCodeMix, in: Proceedings of the 12th Forum for Information Retrieval Evaluation, FIRE '20, 2020.

[3] Pitenis, Z., Zampieri, M. and Ranasinghe, T., Offensive language identification in greek, arXiv preprint arXiv:2003.07459., 2020.

[4] Rosenthal, S., Atanasova, P., Karadzhov, G., Zampieri, M. and Nakov, P., A large-scale semi-supervised dataset for offensive language identification, arXiv preprint arXiv:2004.14454., 2020.

[5] Garain, A., Garain at SemEval-2020 Task 12: Sequence based Deep Learning for Categorizing Offensive Language in Social Media. arXiv preprint arXiv:2009.01195, 2020.

[6] Ahn, H., Sun, J., Park, C.Y. and Seo, J., NLPDove at SemEval-2020 Task 12: Improving Offensive Language Detection with Cross-lingual Transfer, arXiv preprint arXiv:2008.01354, 2020.

[7] Sreelakshmi, K., Premjith, B. and Soman, K.P., Detection of Hate Speech Text in Hindi-English Code-mixed Data. Procedia Computer Science, 171, pp.737-744.

[8] Zampieri, M., Malmasi, S., Nakov, P., Rosenthal, S., Farra, N. and Kumar, R., Predicting the type and target of offensive posts in social media, 2019, arXiv preprint arXiv:1902.09666.

[9] Ranasinghe, T., Zampieri, M., and Hettiarachchi, H. , BRUMS at HASOC 2019: Deep Learning Models for Multilingual Hate Speech and Offensive Language Identification, In FIRE (Working Notes) 2019, pp. 199-207.

[10] B. R. Chakravarthi, Leveraging orthographic information to improve machine translation of under-resourced languages, PhD diss., NUI Galway, 2020.

[11] 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://www.aclweb.org/anthology/2020.sltu-1.25.

[12] 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://www. aclweb.org/anthology/2020.sltu-1.28.

[13] Hochreiter S, Schmidhuber J, Long short-term memory. Neural computation, 1997 Nov 15;9(8):1735-80.

[14] Yang Z, Yang D, Dyer C, He X, Smola A, Hovy E, Hierarchical attention networks for document classification, In Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies 2016 Jun, pp. 1480-1489.