Towards Offensive Language Identification for Dravidian Languages

Siva Sai
Birla Institute of Technology and Science Pilani, India.
f20170779@pilani.bits-pilani.ac.in

Yashvardhan Sharma
Birla Institute of Technology and Science Pilani, India.
yash@pilani.bits-pilani.ac.in

Abstract

Offensive speech identification in countries like India poses several challenges due to the usage of code-mixed and romanized variants of multiple languages by the users in their posts on social media. The challenge of offensive language identification on social media for Dravidian languages is harder, considering the low resources available for the same. In this paper, we explored the zero-shot learning and few-shot learning paradigms based on multilingual language models for offensive speech detection in code-mixed and romanized variants of three Dravidian languages - Malayalam, Tamil, and Kannada. We propose a novel and flexible approach of selective translation and transliteration to reap better results from fine-tuning and ensembling multilingual transformer networks like XLM-RoBERTa and mBERT. We implemented pre-trained, fine-tuned, and ensembled versions of XLM-RoBERTa for offensive speech classification. Further, we experimented with inter-language, inter-task, and multi-task transfer learning techniques to leverage the rich resources available for offensive speech identification in the English language and to enrich the models with knowledge transfer from related tasks. The proposed models yielded good results and are promising for effective offensive speech identification in low resource settings.[1]

1 Introduction

Offensive speech is defined as speech that causes a person to feel upset, resentful, annoyed, or insulted. In recent years, social media such as Twitter, Facebook and Reddit have been increasingly used for the propagation of offensive speech and the organization of hate and offense-based activities (Mandl et al., 2020; Chakravarthi et al., 2020e). In a country like India with multiple native languages, users prefer to use their regional language in their social media interactions (Thavareesan and Mahesan, 2019, 2020a,b). It has also been identified that users tend to use roman characters for texting instead of the native script. This poses a severe challenge for the identification of offensive speech, considering the under-developed methodologies for handling code-mixed and romanized text (Jose et al., 2020; Priyadharshini et al., 2020).

Until a few years ago, hate and offensive speech were identified manually which is now an impossible task due to the enormous amounts of data being generated daily on social media platforms. The need for scalable, automated methods of hate speech detection has attracted significant research from the domains of natural language processing and machine learning. A variety of techniques and tools like bag of words models, N-grams, dictionary-based approaches, word sense disambiguation techniques are developed and experimented with by researchers. Recent developments in multilingual text classification are led by Transformer architectures like mBERT (Devlin et al., 2018) and XLM-RoBERTa (Conneau et al., 2019). An additional advantage of these architectures, particularly XLM-RoBERTa, is that it yields good results even with lower resource languages and this particular aspect is beneficial to Indian languages which do not have properly established datasets. In our work, we focused on using these architectures in multiple ways. However, there is a caveat in directly using the models on the romanized or code-mixed text: the transformer models are trained on languages in their native script, not in the romanized script in which users prefer to write online. We solve this problem by using a novel way to convert the romanized sentences into their native language while preserving their semantic meaning - selective translation and transliteration.

[1] https://github.com/SivaAndMe/TOLIDL_DravidianLangTech_EACL_2021

Proceedings of the First Workshop on Speech and Language Technologies for Dravidian Languages, pages 18–27

This work is an extension of (Sai and Sharma, 2020). Our important contributions are as follows 1) Proposed selective translation and transliteration for text conversion in romanized and code-mixed settings which can be extended to other romanized and code-mixed contexts in any language. 2) Experimented and analyzed the effectiveness of finetuning and ensembling of XLM-RoBERTa models for offensive speech identification in code-mixed and romanized scripts. 3) Investigated the efficacy of inter-language, inter-task, and multi-task transfer learning techniques and demonstrated the results with t-SNE (Maaten and Hinton, 2008) visualizations.

The rest of the paper is organized as follows: In section 2, we discuss related work followed by datasets' description in section 3. Section 4 describes methodology and section 5 discusses about results. Finally, we conclude the paper in section 6.

2 Related works

Two major shared tasks organized in offensive language identification are OffensEval 2019 (Zampieri et al., 2019) and GermEval (Struß et al., 2019). Other tasks related to offensive speech identification include HASOC-19 (Mandl et al., 2019) which dealt with hate speech and offensive content identification in Indo-European languages, TRAC-2020 (Kumar et al., 2020), which dealt with aggression identification in Bangla, Hindi and English. While HASOC-19 and TRAC 2020 dealt with offensive speech identification in Indian languages of Bangla and Hindi, HASOC-Dravidian-CodeMix - FIRE 2020 (Chakravarthi et al., 2020b) is the first shared task to conduct offensive speech identification task in Dravidian languages.

Researchers used a wide variety of techniques for the identification of offensive language. Recently, (Ranasinghe and Zampieri, 2020)(work published after the first version of this paper was submitted) used the XLM-RoBERTa model for offensive language identification in Bengali, Hindi, and Spanish. They show that the XLM-R model beats all other previous approaches for offensive language detection. (Saha et al., 2019), used the LGBM classifier on top of the combination of multilingual BERT and LASER pre-trained embeddings in HASOC-19. (Mishra and Mishra, 2019) fine-tuned monolingual and multilingual BERT based network models to achieve good results in identifying hate speech. (Risch and Krestel, 2020) used an ensemble of BERT models with different random seeds for aggression detection on social media text, which is the inspiration behind our ensembling strategy with XLM-RoBERTa models. There has been less research on text classification in Dravidian languages and there is no research in offensive speech identification for Dravidian languages so far. (Thomas and Latha, 2020) uses a simple LSTM model for sentiment analysis in the Malayalam language. Our work adresses this gap of less research in offensive speech identification methods for Dravidian languages and the systems proposed can be extended to other Indian and foreign languages as well.

3 Datasets

The datasets used in this work were taken from three competitions - HASOC-Dravidian-CodeMix - FIRE 2020 (Chakravarthi et al., 2020a,c,b,d; Chakravarthi, 2020) (henceforth HDCM), Sentiment Analysis for Dravidian Languages in Code-Mixed Text (Chakravarthi et al., 2020a,d)(henceforth SADL) and Offensive Language Identification in Dravidian Languages (Chakravarthi et al., 2021; Hande et al., 2020; Chakravarthi et al., 2020d,a)(henceforth ODL). As a part of HDCM, there were two binary classification tasks, with the second task having two subtasks. The objective of all of the tasks was the same: given a Youtube comment, classify it as offensive or not offensive. But the format and language of data provided to different tasks were different: Code-mixed Malayalam for Task-1(henceforth referred to as Mal-CM), Tanglish for Task- 2a(henceforth referred to as Tanglish), and Manglish for Task-2b(henceforth referred to as Manglish). From ODL, we used the Kanglish dataset provided. Originally, the task proposed in ODL is a multi-class classification problem with multiple offensive categories. However, to maintain uniformity among the datasets, we divided the classes into two - offensive and not offensive. Offensive-Targeted-Insult-Individual, Offensive-Targeted-Insult-Group, Offensive-Untargeted, and Offensive-Targeted-Insult-Other classes in the dataset are renamed as offensive, and all the non-Kannada posts are removed from the dataset. As far as SADL is concerned, we used the given sentiment analysis datasets as helper datasets in inter-task transfer learning technique(4.5). OLID (Zampieri et al., 2019) is a dataset for offensive language iden-

Task	Train		Dev	Test
	OFF	NOT		
Mal-CM	567	2633	400	400
Tanglish	1980	2020	-	940
Manglish	1952	2048	-	951
Kanglish	1151	3544	586	778
OLID	4400	8840	-	860

Table 1: Statistics of the offensive speech datasets used in this work.

tification in the English language and we used it as a helper dataset for inter-language transfer learning technique (4.4). It consists of 14,100 tweets as stated before; each tweet is annotated as offensive or not-offensive(subtask-A). The statistics of the offensive speech datasets used in this work are provided in table 1.

4 System description

4.1 Preprocessing

Further, we have done following pre-processing on text for all the datasets: a) Lower case the romanized words. This step is not performed for words written in Malayalam script in Mal-CM as there is no such casing used in Malayalam script. b) Remove emojis from text. c) Remove all special characters, numbers and punctuation. d) Remove user mentions as they generally do not carry any semantic meaning.

4.2 Selective translation and transliteration(STT)

This is a novel idea we have used to get a proper representation of text in the native script for the final neural architecture training. The pseudo-code for the proposed algorithm is given in Algorithm 1. The primary need for this step is as follows: Recent advancements in state-of-the-art multilingual NLP tasks are led by Transformer architectures like mBERT and XLM-RoBERTa which are trained on multiple languages in the native script but not in the romanized script[2]. Hence to reap better results by fine-tuning these architectures, the text is to be in a native script(for example, Tamil text in Tamil script).

To convert text into the native script, we cannot rely on neural translation systems, particularly in tweets where users tend to write informally using multiple languages. Also, translating romanized non-English language words into that particular language does not make any sense in our context. For example, translating the word "Maram"(which means tree in Tamil) directly into Tamil would seriously affect the entire sentence's semantics in which the word is present because "Maram" will be treated as an English word. In many cases, valid translations from English to a non-English language would not be available for words.

So, as a solution to this problem, we propose selective transliteration and translation of the text. In effect, this process of conversion of romanized or code-mixed text(for example, Tanglish) is to transliterate the romanized native language(Tamil) words in the text into Tamil and translate the English words in the text into Tamil selectively. The segregation of English words and native language words in a given sentence is done using a big corpus of English words from NLTK-corpus[3]. The idea of this selective conversion is based on the observation that in romanized native language comments(like Tanglish), users tend to use English words only when they can convey the meaning better with the English word or when the corresponding native language word is not much used in regular conversations. For example, the word "movie" is preferred by Tamil-users than its corresponding Tamil word.

The translation of words is done using Google Translate API[4], and transliteration is done with the help of BrahmiNet API[5]. The detection of language script is carried out with the help of langdetect API[6].

4.2.1 Variation of STT algorithm

In some tweets, the direct translation of intermediate English words present in them can affect the semantics negatively, and it may be better to keep them as they are. So, we have experimented with this variation of the STT algorithm - only transliterate the non-English words into their corresponding language and keep the English words as they are. We observed that the cross-lingual nature of the XLM-RoBERTa model can efficiently handle these types of mixed script sentences.

[2]except few languages like Telugu, Tamil and Urdu for XLM-RoBERTa.

[3]https://www.nltk.org/

[4]https://pypi.org/project/googletrans/

[5]http://www.cfilt.iitb.ac.in/brahminet/static/rest.html

[6]https://pypi.org/project/langdetect/

Algorithm 1: Algorithm for Selective Translation and Transliteration of code mixed and romanized languages

Input : Romanized or Code-mixed text T and desired native language for the final script L

Output : Text in native script

```
Initialization: EngWords = Set of all english words
words = splitSentIntoWords(T)
LOOP Process
for i=0 to len(words) do
    word = words[i]
    if(detectLanguageScript(word)==L) then
        continue
    else if(word in EngWords) then
        words[i] = translate(word,L)
    else
        words[i] = transliterate(word,L)
    endif
end for
return joinWordsToSent(words)
```

4.3 Models

Recent studies show that pre-trained word embeddings and fine-tuning of state-of-the-art Transformer architectures show better performance in text classification compared to classical machine learning approaches like N-gram features with bag of words models (Saha et al., 2019). So we directed our entire focus on using the word-embeddings of transformer architectures both pre-trained and fine-tuned for text classification. The text obtained using selective translation and transliteration is used in further steps.

4.3.1 XLM-RoBERTa

XLM-RoBERTa (Conneau et al., 2019) is a large multilingual model trained on 2.5TB of CommonCrawl data in 100 different languages. It shows improved performance on low-resource languages and outperforms other transformer models like mBERT on cross-lingual benchmarks. The model gives good performance on multilingual datasets without losing the competitive edge on monolingual benchmarks.The base version of the XLM-RoBERTa model consists of 12 hidden layers, 250k parameters, and 12 attention heads.

4.3.2 Pre-trained word embeddings

Transfer learning using pre-trained word embeddings is proved to be useful for offensive speech detection tasks (Saha et al., 2019). So we experimented with XLM-RoBERTa pre-trained embeddings in our work. The pre-trained XLM-R model takes text as input and outputs feature vector of size 1024 for each token in the sentence. We take the average of the feature vectors for all tokens as the final feature vector for the entire sentence.

The pretrained feature vectors of size 1024 are given as input to classical classification algorithms like Logistic Regression. We performed an exhaustive classifier search among 16 classifiers like DecisionTreeClassifier, XGBoostClassifier, etc., to find the classifier that performs better for each task. Our observations show that Logistic Regression outperforms others for Tanglish and Manglish datasets, whereas MLP classifier shows better performance for Mal-CM. We also experimented with a neural network classifier on top of the word embeddings in place of classical algorithms. However, it did not improve the performance much. We have used the Pytorch[7] framework to obtain pre-trained XLM-R embeddings and Sklearn[8] for classifiers.

4.3.3 Fine-tuning Transformer architectures

When we extract features from the pre-trained model, we are using the base model as it is. However, we can fine-tune the base model to customize on our dataset to improve performance. We used Multilingual BERT(uncased) (Devlin et al., 2018), XLM-RoBERTa(both base and large versions) for fine-tuning. For both mBERT and XLM-R, the final hidden state of first token [CLS] (which represents the entire sentence) is fed to a softmax layer for text classification. We performed minimal hyperparameter tuning. Early stopping with a patience of 10 is used targeting the f1-weighted score. A maximum sequence tokens length of 70 is used for all the models based on the observation that around 95% of posts have lesser than 70 tokens. AdamW optimizer with a weight decay of 0.01 is utilized. Cross entropy loss is used as the loss function. We have evaluated the model once for every 100 batches during fine-tuning with 50 as maximum number of epochs. It implies that the model is evaluated once for every 1.25 epochs. All the experiments in this work are performed using

[7] https://pytorch.org/
[8] https://scikit-learn.org/

Google Colab GPU runtime[9].

4.3.4 Ensembling Transformer architectures

The instability and variance of the transformer architectures' performance is the motivation behind this ensembling strategy (Risch and Krestel, 2020). (Devlin et al., 2018) show that the performance(accuracy score) of the BERT model on small datasets, such as the Microsoft Research Paraphrase Corpus (MRPC), varies between 84% and 88%. In our experiments with XLM-R base models, we observed a similar pattern:± 5% F1-weighted score for Mal-CM and Manglish, and ±4% F1-weighted score for Tanglish on validation data. This variance can be created by slight changes in hyperparameters(random seed particularly) and training data. The random seed of a model affects the initialization of weights of the final classification layer. Furthermore, change in random seed while splitting the data into train and validation sets decide which samples go into each of them. This also affects the ordering of the samples in a particular set.

We experimented with 10 different random seeds on XLM-R base model. (Risch and Krestel, 2020) reports that 15 is the optimal number of BERT models for ensembling and that the performance plateaus above that number. In our experiments with XLM-R base, we observed that ensembling 10 models is optimal, where each model corresponds to a different random seed. All other hyperparameters are kept the same as that of 4.3.3 for ensembling. We used soft-majority voting to combine predictions of these ten models. Soft majority voting simply adds the probabilities of each class from all the models and chooses the one with highest probability as the predicted class.

Apart from ensembling of same XLM-RoBERTa models, we also experimented with ensembles of different models : XLM-RoBERTa base + XLM-RoBERTa large and XLM-RoBERTa base + mBERT to analyse the effect of diversity of architectures on the performance.

4.4 Inter-language transfer learning

The intuition behind this transfer learning technique is as follows: the patterns of expression of offensive content can have similarities across different languages. Hence, fine-tuning the neural network first on a large scale dataset and using its knowledge(weights) to fine-tune the model on another smaller or under-resourced language dataset can improve the performance. The size of OLID dataset is large as compared to the datasets that we are targeting in our work. We first fine-tune the XLM-RoBERTa model (base version) on the OLID dataset. After this first fine-tuning step, the weights of all layers except the pre-final fully connected layer and the final softmax layer are used for initializing the XLM-RoBERTa model, which is to be fine-tuned for offensive speech classification in a Dravidian language.

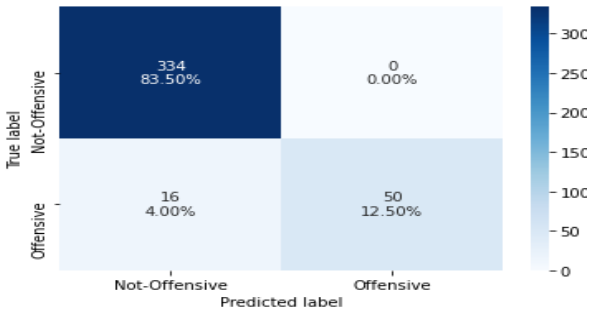

Figure 1: Mal-CM - STT variant XLM-R - confusion matrix

4.5 Inter-task transfer learning

In this type of transfer learning technique, the knowledge learned by fine-tuning a language model with a particular objective is transferred to another language model for achieving a different objective. In our work, we keep the language of the datasets for the two tasks the same. However, it can be different, as well. Firstly, we fine-tune the XLM-RoBERTa model for multi-label multi-class sentiment analysis in a particular code-mixed and romanized language and use the final weights of the neural network layers for initializing weights for another XLM-RoBERTa model, which is used for offensive speech identification in the same code-mixed language. The weights of the final softmax layer are not transferred, as it is evident that two different tasks are being dealt with here. We used this technique for Tanglish and Manglish datasets because the corresponding sentiment analysis datasets are available only for these languages. We used datasets from SADL for the initial fine-tuning. We show that the first model's knowledge in dealing with code-mixed languages is helpful for the second model.

[9] https://colab.research.google.com/

Task	XLMR-pretrained+clf	mBERT	XLMR-B	XLMR-B+ mBERT	XLMR-B + XLMR-L
Mal-CM	0.88	0.91	**0.93**	**0.93**	0.93
Tanglish	0.78	0.83	0.86	**0.88**	0.85
Manglish	0.67	0.60	**0.69**	**0.69**	**0.69**
Kanglish	0.80	0.81	**0.82**	**0.82**	**0.82**

Table 2: Results on Validation data - F1-Weighted score

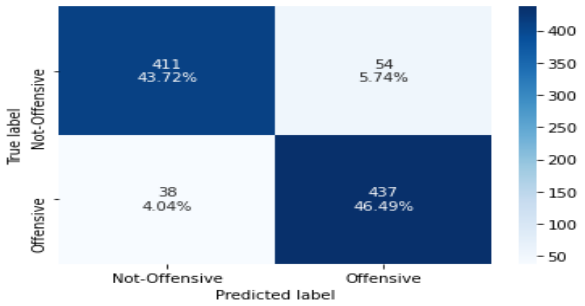

Figure 2: Tanglish - Ensemble of XLMR models - confusion matrix

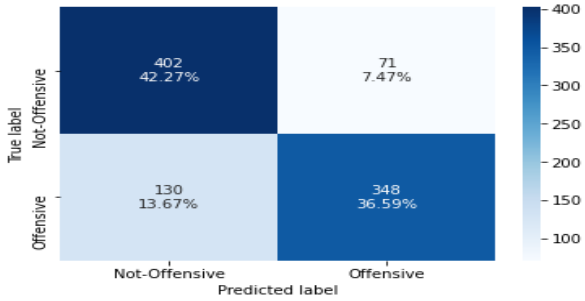

Figure 3: Manglish - Ensemble of XLMR models - confusion matrix

4.6 Multi-task transfer learning(MTL)

As the name implies, the neural network learns two or more tasks simultaneously in this type of transfer learning technique. The intuition behind this technique is that the learning of the model is enhanced by dealing simultaneously with multiple tasks due to the inherent similarities and differences present in data and the learning approach. (Caruana, 1998) states that the goal of MTL is to improve the generalization capability of the model by leveraging domain specific knowledge present in the training data of related tasks. In our work, we experimented with MTL on the Kanglish dataset, with two related tasks - fine-grained and coarse-grained identification of offensive speech. To implement MTL, two separate fully connected layers are added on top of the same XLM-R model for final classification. The loss is an aggregate of the individual losses of two tasks. In our context, the goal is to improve the coarse-grained classification(offensive or not-offensive) with the help of fine-grained classification(not-offensive or offensive-targeted-insult-individual or offensive-targeted-insult-group or offensive-untargeted or offensive-targeted-insult-other).

5 Results and discussion

To test the performance of our proposed models, we had evaluated them on validation data before the test set was made available by the concerned task organizers. When unlabeled test data was released, we used the above-mentioned validation data as dev set to develop the model, and final predictions are made using these models. Ensemble of XLM-RoBERTa models is not experimented on dev data due to computational limitations. Nevertheless, we verified the better performance of Ensemble models with few random seeds and directly used them for final training.

5.1 Results on validation data

We created stratified dev sets from training data for Tanglish and Manglish. The dev sets for Mal-CM and Kanglish are provided by the organizers of HDCM and ODL respectively. And these dev sets are used for final internal valuation as test sets. A 10% of training data was used as dev data for training for all the tasks. The results are shown in table 2. In the case of pre-trained embeddings, clf(classifier) used is Logistic Regression for Tanglish, Kanglish and Manglish tasks and MLP for Mal-CM. Because XLM-RoBERTa is also trained on Tamil dataset in roman script, we experimented with directly feeding the pre-processed Tanglish text to the model without STT. But, the performance is significantly lower than the model which follows entire pipeline.

It can be observed from the table 2 that fine-tuning models gave better results than directly using off-the-shelf embeddings. But for Manglish and Kanglish, pre-trained embeddings come closer to fine-tuned models in performance. The superior

Model	Mal-CM	Tanglish	Manglish	Kanglish
STT + XLMR	0.92	0.90	0.73	0.82
STT variant + XLMR	**0.96**	0.90	0.75	**0.82**
Ensemble of XLMR models	0.95	**0.90**	**0.79**	–
Inter-language Transfer Learning	0.94	0.87	0.75	0.81
Inter-task Transfer Learning	0.94	0.90	0.71	–
Multi-task Transfer Learning	–	–	–	0.78

Table 3: Results on test data - F1-Weighted

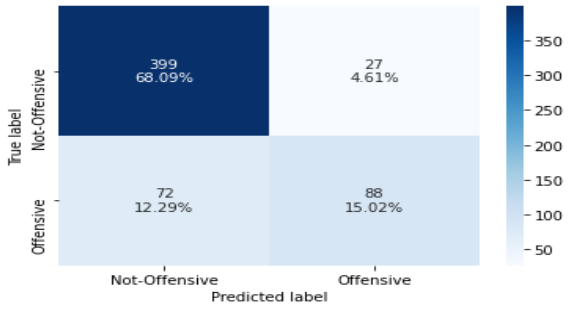

Figure 4: Kanglish - STT XLMR - confusion matrix

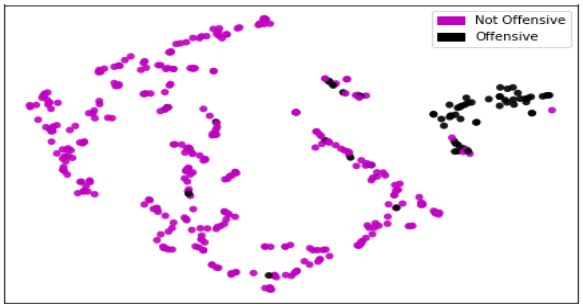

Figure 5: t-SNE visualization for Mal-CM on test data

performance of XLMR-base model in all the tasks can also also be inferred. This is the reason, we chose XLMR-base for ensembling strategy.

5.2 Results on test data

We submitted the models proposed for Mal-CM, Tanglish, and Manglish to HDCM and the results were encouraging. Our submissions topped two out of three tasks(Mal-CM and Tanglish) and stood second in the third task lagging the top model only by 0.01 points(Manglish). We can see that the test F1-weighted scores are slightly better than validation results for Mal-CM and Tanglish tasks (table 3). And the results on test data are significantly higher than results on dev data for Manglish task(8% higher). It can be inferred that performance scores for Manglish are significantly less than those of Tanglish task(10% lower). We attribute this to agglutinative and inflectional nature of Malayalam language.

STT v/s STT variant

From the results, it can also be observed that the XLM-R model using STT variant algorithm performed better than(or at par with) XLM-R model using STT on all datasets. We attribute this to the efficiency of XLM-R model to handle cross-lingual sentences(Conneau et al., 2019). Moreover, in some instances, the STT may give bad translations. For instance, when the sentence is entirely in English, STT algorithm translates every word, affecting the meaning of the sentence because, in general, the word-to-word translation of a sentence will not give the correct meaning in the target language. This is a drawback of STT. But the assumption that most of the corpus text is in code-mixed and romanized format reduces the issues from this drawback. Notably, the STT variant based model performs even better than the ensemble of XLMR models(which is computationally intensive) on some datasets.

Inter-language and inter-task transfer learning

The initialization of weights from OLID based XLM-R model(inter-language transfer learning technique) helped Mal-CM and Manglish compared to the basic XLM-R model(with STT) while decreasing the performance for Tanglish. The initialization of weights from SADL based XLM-R model(inter-task transfer learning technique) increased the performance for Mal-CM and decreased the performance for Manglish compared to the basic XLM-R model.

Although the inter-language and inter-task transfer learning did not improve the results largely, we observed that the initial validation scores while training are significantly higher for those with SADL or OLID based initialization compared to a model initialized with random weights. We attribute this to the initial transfer of useful knowledge from the SADL and OLID tasks.

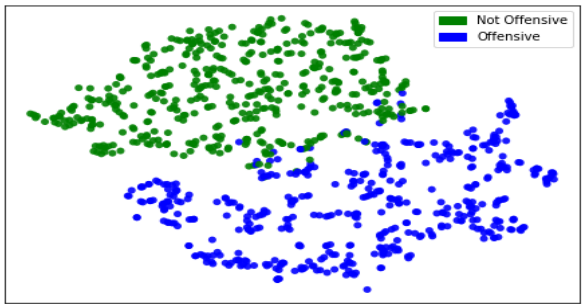

Figure 6: t-SNE visualization for Tanglish on test data

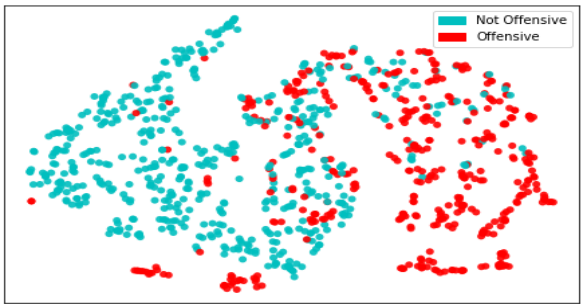

Figure 7: t-SNE visualization for Manglish on test data

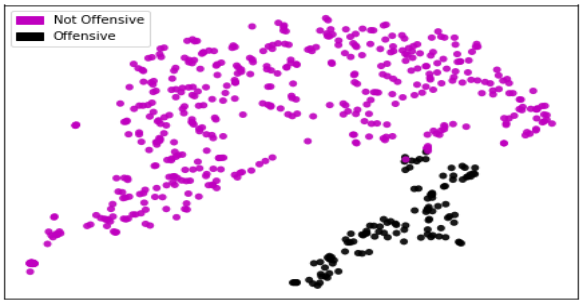

Figure 8: t-SNE visualization for Kanglish on test data

Multi-task transfer learning
The notable decrease of F1-weighted score for multi-task learning based XLM-R model for Kanglish corroborates with the findings of (Sun et al., 2019). Although the task of coarse-grained classification (as a part of MTL) did not gain anything, the fine-grained classification has substantial gains in terms of performance. This second task achieved an F1-weighted score of 72.8% for fine-grained classification with five classes, which is fair enough considering that the best F1-weighted score for two-way classification is 82%(as shown in table 3).

We could not perform certain experiments(refer table 3) like SADL-based weight initialization for Kanglish and MTL with datasets other than Kanglish because of lack of suitable datasets.

t-SNE Visualizations and Confusion matrices
We projected the 768-dimensional feature vectors obtained from the final layer of XLM-RoBERTa model(with STT) onto a two-dimensional space using t-SNE algorithm (Maaten and Hinton, 2008) for all the tasks on test data (refer figures 5,6,7 and 8). The distinction between two clusters in all of the t-SNE visualizations are clearly in line with the results given in table 3. For example, the clusters for Tanglish(figure 6) and Mal-CM(figure 5) are almost well segregated, but the clusters for Manglish(figure 7) are overlapping. We have also provided the confusion matrices for best performing models on each dataset - STT variant XLMR model for Mal-CM(figure 1), ensemble of XLMR models for Tanglish(figure 2) and Manglish(figure 3), and STT XLMR model for Kanglish(figure 4).

6 Conclusion

In this paper, we have presented various techniques and neural network models to identify offensive language in social media text for code mixed and romanized Dravidian languages. A novel technique of selective translation and transliteration is proposed to deal with code-mixed and romanized offensive speech classification in Dravidian languages. This technique is flexible and can be extended to other languages as well. For classification, classical classifiers on top of pre-trained embeddings, fine-tuned XLM-RoBERTa models, and an ensemble of XLM-RoBERTa models are used. We experimented with different transfer learning techniques to leverage the offensive speech datasets from resource-rich languages. Our work also points to the usefulness of Transformer architectures, particularly XLM-RoBERTa, for low resource languages like Tamil and Malayalam. Our proposed models show an average performance of 85% F1-weighted score across all datasets.

7 Acknowledgement

The authors would like to convey their sincere thanks to the Department of Science and Technology (ICPS Division), New Delhi, India, for providing financial assistance under the Data Science (DS) Research of Interdisciplinary Cyber Physical Systems (ICPS) Programme [DST /ICPS /CLUSTER /Data Science/2018/Proposal-16:(T-856)] at the department of computer science, Birla Institute of Technology and Science, Pilani, India.

References

R Caruana. 1998. Multitask learning. autonomous agents and multi-agent systems.

Bharathi Raja Chakravarthi. 2020. *Leveraging orthographic information to improve machine translation of under-resourced languages*. Ph.D. thesis, NUI Galway.

Bharathi Raja Chakravarthi, Navya Jose, Shardul Suryawanshi, Elizabeth Sherly, and John Philip McCrae. 2020a. 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)*, pages 177–184, Marseille, France. European Language Resources association.

Bharathi Raja Chakravarthi, M Anand Kumar, John Philip McCrae, Premjith B, Soman KP, and Thomas Mandl. 2020b. 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*.

Bharathi Raja Chakravarthi, M Anand Kumar, John Philip McCrae, Premjith B, Soman KP, and Thomas Mandl. 2020c. Overview of the track on "HASOC-Offensive Language Identification-DravidianCodeMix". In *Proceedings of the 12th Forum for Information Retrieval Evaluation*, FIRE '20.

Bharathi Raja Chakravarthi, Vigneshwaran Muralidaran, Ruba Priyadharshini, and John Philip McCrae. 2020d. 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)*, pages 202–210, Marseille, France. European Language Resources association.

Bharathi Raja Chakravarthi, Ruba Priyadharshini, Navya Jose, Anand Kumar M, Thomas Mandl, Prasanna Kumar Kumaresan, Rahul Ponnusamy, Hariharan V, Elizabeth Sherly, and John Philip McCrae. 2021. 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.

Bharathi Raja Chakravarthi, Ruba Priyadharshini, Vigneshwaran Muralidaran, Shardul Suryawanshi, Navya Jose, Elizabeth Sherly, and John P. McCrae. 2020e. Overview of the Track on Sentiment Analysis for Dravidian Languages in Code-Mixed Text. In *Forum for Information Retrieval Evaluation*, FIRE 2020, page 21–24, New York, NY, USA. Association for Computing Machinery.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Adeep Hande, Ruba Priyadharshini, and Bharathi Raja Chakravarthi. 2020. 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*, pages 54–63, Barcelona, Spain (Online). Association for Computational Linguistics.

Navya Jose, Bharathi Raja Chakravarthi, Shardul Suryawanshi, Elizabeth Sherly, and John P. McCrae. 2020. A Survey of Current Datasets for Code-Switching Research. In *2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS)*, pages 136–141.

Ritesh Kumar, Atul Kr. Ojha, Bornini Lahiri, Marcos Zampieri, Shervin Malmasi, Vanessa Murdock, and Daniel Kadar, editors. 2020. *Proceedings of the Second Workshop on Trolling, Aggression and Cyberbullying*. European Language Resources Association (ELRA), Marseille, France.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605.

Thomas Mandl, Sandip Modha, Anand Kumar M, and Bharathi Raja Chakravarthi. 2020. Overview of the HASOC Track at FIRE 2020: Hate Speech and Offensive Language Identification in Tamil, Malayalam, Hindi, English and German. In *Forum for Information Retrieval Evaluation*, FIRE 2020, page 29–32, New York, NY, USA. Association for Computing Machinery.

Thomas Mandl, Sandip Modha, Prasenjit Majumder, Daksh Patel, Mohana Dave, Chintak Mandlia, and Aditya Patel. 2019. Overview of the hasoc track at fire 2019: Hate speech and offensive content identification in indo-european languages. In *Proceedings of the 11th Forum for Information Retrieval Evaluation*, pages 14–17.

Shubhanshu Mishra and Sudhanshu Mishra. 2019. 3idiots at hasoc 2019: Fine-tuning transformer neural networks for hate speech identification in Indo-European languages. In *FIRE (Working Notes)*, pages 208–213.

Ruba Priyadharshini, Bharathi Raja Chakravarthi, Mani Vegupatti, and John P. McCrae. 2020. Named Entity Recognition for Code-Mixed Indian Corpus using Meta Embedding. In *2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS)*, pages 68–72.

Tharindu Ranasinghe and Marcos Zampieri. 2020. Multilingual offensive language identification with cross-lingual embeddings. *arXiv preprint arXiv:2010.05324*.

Julian Risch and Ralf Krestel. 2020. Bagging bert models for robust aggression identification. In *Proceedings of the Second Workshop on Trolling, Aggression and Cyberbullying*, pages 55–61.

Punyajoy Saha, Binny Mathew, Pawan Goyal, and Animesh Mukherjee. 2019. Hatemonitors: Language agnostic abuse detection in social media. *arXiv preprint arXiv:1909.12642*.

Siva Sai and Yashvardhan Sharma. 2020. Siva@HASOC-Dravidian-CodeMix-FIRE-2020: Multilingual Offensive Speech Detection in Code-mixed and Romanized text. FIRE.

Julia Maria Struß, Melanie Siegel, Josef Ruppenhofer, Michael Wiegand, Manfred Klenner, et al. 2019. Overview of germeval task 2, 2019 shared task on the identification of offensive language.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019. How to fine-tune bert for text classification? In *China National Conference on Chinese Computational Linguistics*, pages 194–206. Springer.

Sajeetha Thavareesan and Sinnathamby Mahesan. 2019. Sentiment Analysis in Tamil Texts: A Study on Machine Learning Techniques and Feature Representation. In *2019 14th Conference on Industrial and Information Systems (ICIIS)*, pages 320–325.

Sajeetha Thavareesan and Sinnathamby Mahesan. 2020a. Sentiment Lexicon Expansion using Word2vec and fastText for Sentiment Prediction in Tamil texts. In *2020 Moratuwa Engineering Research Conference (MERCon)*, pages 272–276.

Sajeetha Thavareesan and Sinnathamby Mahesan. 2020b. Word embedding-based Part of Speech tagging in Tamil texts. In *2020 IEEE 15th International Conference on Industrial and Information Systems (ICIIS)*, pages 478–482.

Merin Thomas and CA Latha. 2020. Sentimental analysis of transliterated text in malayalam using recurrent neural networks. *Journal of Ambient Intelligence and Humanized Computing*, pages 1–8.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019. Semeval-2019 task 6: Identifying and categorizing offensive language in social media (offenseval). *arXiv preprint arXiv:1903.08983*.