LA-SACo: A Study of Learning Approaches for Sentiments Analysis in Code-Mixing Texts

F. Balouchzahi
Department of Computer Science,
Mangalore University,
Mangalore - 574199, India
frs_b@yahoo.com

H. L. Shashirekha
Department of Computer Science,
Mangalore University,
Mangalore - 574199, India
hlsrekha@gmail.com

Abstract

Sentiments are usually written using a combination of languages such as English which is resource rich and regional languages such as Tamil, Kannada, Malayalam, etc. which are resource poor. However, due to technical constraints, many users prefer to pen their opinions in Roman script rather than using their native scripts. These kinds of texts written in two or more languages using a common language script or different language scripts are called code-mixing texts. Code-mixed texts are increasing day-by-day with the increase in the number of users depending on various online platforms. Analyzing such texts pose a real challenge for the researchers. In view of the challenges posed by the code-mixed texts, this paper describes three proposed models namely, SACo-Ensemble, SACo-Keras, and SACo-ULMFiT using Machine Learning (ML), Deep Learning (DL), and Transfer Learning (TL) approaches respectively for the task of sentiments analysis in Tamil-English and Malayalam-English code-mixed texts. The results illustrate that SACo-Ensemble with weighted F1-scores of 0.62 and 0.72 on Tamil-English and Malayalam-English language pairs respectively outperformed other proposed models.

1 Introduction

Feelings, opinions, or reviews of customers in online shops or social media such as YouTube, Facebook, WhatsApp, Instagram, Twitter, etc. are called sentiments. Feedback or opinion about whatever is available on the internet can also be seen as sentiments (Thavareesan and Mahesan, 2019, 2020a,b). Increase in the number of people using social media and online platforms have given rise to increasing amount of text data in general and sentiments or opinions in particular. Sentiments or reviews posted by users affect the popularity of a post, and video in social media or a product in online shops (Chakravarthi and Muralidaran, 2021; Chakravarthi et al., 2021; Suryawanshi and Chakravarthi, 2021). For example, positive sentiments could be a reason for making a post/product popular whereas negative reviews could go to the extent of rejecting or discarding a particular post/product. The sentiments extracted from users' posts and customers feedbacks tend to be valuable information not only for a particular user to be encouraged to watch a movie or buy a product but also for many social media companies and online shops or even movie makers, automakers etc. to improve their weaknesses and strengths. For instance, the negative feedbacks and sentiments gained from users about confusion in the new policy of WhatsApp about sharing data and location, and business messaging has substantially reduced the popularity of this platform[1]. Hence, Sentiment Analysis (SA), the task of automatically analyzing these sentiments or reviews posted by the users for the proper identification and classification of sentiments is becoming crucial these days (Balouchzahi and Shashirekha, 2020a).

Researchers motivated to explore SA have developed many tools and techniques to analyze sentiments written in rich resource languages such as English, Spanish, etc. But, most of the people would like to express opinions in their native language in native script or other language script due to freedom to use any language and any script in online platforms (Chakravarthi et al., 2018). For example, South Indian people may use Dravidian languages (Telugu, Tamil, Malayalam, and Kannada) to post their sentiments (Chakravarthi et al., 2020b). In the 2nd century BCE, the Dravidian languages were first attested to as a Tamili [2] script inscribed

[1] https://indianexpress.com/article/technology/tech-news-technology/whatsapp-privacy-policy-update-delay-backlash-7149456/

[2] also called Damili or Tamil-Bhrami

Proceedings of the First Workshop on Speech and Language Technologies for Dravidian Languages, pages 109–118
April 20, 2021 ©2021 Association for Computational Linguistics

on the cave walls of Tamil Nadu's Madurai and Tirunelveli districts. Telugu, Tamil, Kannada, and Malayalam are the Dravidian languages with the most speakers (in descending order of number of speakers), of which Tamil have long literary traditions from 600 BCE. Over 55% of the epigraphical engravings (around 55,000) found by the Archeological Survey of India are in the Tamil language. However due to technical constraints and difficulty in using Indian languages scripts users usually use Roman script instead of their native scripts along with English words (Balouchzahi and Shashirekha, 2020a; Chakravarthi, 2020b). Mixing two or more languages in a text using a common language script or different languages scripts is called Code-Mixing and code-mixed texts are increasing with the popularity of social media and online shopping (Ansari and Govilkar, 2018; Chakravarthi, 2020a).

SA task is more challenging with code-mixed texts since the sentiments or reviews written in words of different languages increases the complexity of analyzing such texts due to code-mixing at various levels such as words, phrases, and sentences (Lal et al., 2019; Priyadharshini et al., 2020; Jose et al., 2020). Further, as there are no rules governing the formation of code-mixed texts such texts usually contain incomplete and incorrect sentences, short forms of words and words with repetitive letters, e.g. hellooooo, aaaaaa, soooorrryyyy (Puranik et al., 2021; Hegde et al., 2021; Yasaswini et al., 2021; Ghanghor et al., 2021b,a). These complexities give rise to building new features structures such as combined vocabulary and grammar of different languages which cannot be handled by conventional SA models as they fail to capture the meaning of the sentences in code-mixed text (Choudhary et al., 2018).

In view of the complexities and challenges of code-mixed texts, this paper describes the three proposed SA models namely, SACo-Ensemble, SACo-Keras, and SACo-ULMFiT using Machine Learning (ML), Deep Learning (DL), and Transfer Learning (TL) approaches respectively for the task of SA in Tamil-English (Ta-En) and Malayalam-English (Ma-En) code-mixed texts. While SACo-Ensemble and SACo-Keras models utilize a set of features comprised of char sequences, BytePair Encoded words, and syntactic ngrams, SACo-ULMFiT model uses the knowledge obtained from a source model called Language Model in training the sentiment analysis classifier as target model.

2 Related Work

Several datasets, tools, and techniques have been developed to deal with code-mixed data of different language pairs for various tasks including SA, language identification, POS tagging, NER, etc. Some of recent ones are given below: The objective of shared task "Sentiment Analysis of Dravidian Languages in Code-Mixed Text"[3] (Chakravarthi et al., 2020c) was to develop and test SA models for Ta-En and Ma-En code-mixed datasets created by (Chakravarthi et al., 2020b) (Chakravarthi et al., 2020a). This task received 32 and 38 submissions for Ta-En and Ma-En respectively and a given sentiment was categorized into one of five categories, namely, Positive, Negative, Unknown_state, Mixed-Feelings, and Other_languages. Overall results based on weighted F1-score illustrates that there was very tough and close competition among the participating teams. Differences between the weighted F1-score of first and fourth rank of Ta-En task and first and sixth rank of Ma-En task (our team MUCS) (Balouchzahi and Shashirekha, 2020a) are only 0.03 and 0.05 respectively. Therefore, two top submissions along with our SACo model (Balouchzahi and Shashirekha, 2020a) are used for comparison in this study and the same are described below: (Sun and Zhou, 2020) presents a XLM-Roberta model for SA which uses extracted output of top hidden layers and feed them to concatenated Convolution Neural Networks (CNN). This model which is able to extract the semantic information from texts obtained first ranks for both Ta-En and Ma-En code-mixed texts with a weighted F1-score of 0.65 and 0.74 for Ta-En and Ma-En code-mixed texts respectively.

(Ou and Li, 2020) have used XLM-Roberta pre-trained multi-language models and K-folding method to ensemble them for solving the SA problem of multilingual code-mixed texts. They obtained 0.63 and 0.74 weighted F1-score and third and first ranks on Ta-En and Ma-En code-mixed texts respectively.

(Balouchzahi and Shashirekha, 2020a) proposed SACo-HVC, a hybrid model that ensembles the DL and ML models for SA of code-mixed texts. They train a Multi-Layer Perceptron (MLP) classifier with a combination of traditional char and word n-grams, Multinomial Naïve Bayes (MNB) classifier on Skipgram word vector generated from

[3] https://dravidian-codemix.github.io/2020/index.html

the training set, and also BiLSTM networks on subWords embedding generated from the training set. Using majority voting of predictions they obtained 0.62 and 0.68 weighted F1-score and forth and sixth ranks on Ta-En and Ma-En respectively.

3 Methodology

Using three different learning approaches, we propose three models namely, SACo-Ensemble - a ML approach, SACo-Keras - a DL approach and SACo-ULMFiT – a TL approach for the SA of Ta-En and Ma-En code-mixed texts. The first two models are trained on vectors generated using Count Vectorizer[4] from a feature set of char sequences, Byte-Pair Encoding (BPEmb) (Heinzerling and Strube, 2017) encoded words and syntactic n-grams. Subsection 3.1 gives details of feature engineering module for ML and DL approaches. Details of the proposed models are presented in Section 3.2.

3.1 Feature Engineering Module

The feature engineering module is responsible to prepare features for the proposed ML and DL approaches. This module will receive dataset as input and preprocess it by converting emojis to corresponding text (using emoji library[5]), removing punctuations, words of length less than 2, unwanted characters (such as !()-[];:",¡¿./?$=% +@*_ ', etc.) and converting the text to lowercase. Then the following features are extracted from the remaining text:

- **Char sequences:** of length 2 to 6 are extracted from every sentence using everygrams function from NLTK library. For example, given a sentence "yuvanvera level ya" in Ta-En code-mixed text, "yu, uv, va, an, n_, _v, ve, er, ra, a_, _l, le, ev, ve, el, l_, _y, ya, yuv, uva, van, an_, _ve, ver, era, ra_, _le, lev, eve, vel, el_, _ya, yuva, uvan, van_, _ver, vera, era_, _lev, leve, evel, vel_, yuvan, uvan_, _vera, vera_, _leve, level, evel_, yuvan_, _vera_, _level, level_" will be generated as features.

- **BPEmb subWords:** is a collection of pre-trained subWords embeddings for 275 languages trained on Wikipedia texts in their own scripts. In this work, a embedding of vocabulary size 10,000 is chosen for English language to encode code-mixing text and extract subWords from sentences. For example, given a sentence "Indiayale friend" in Ma-En code-mixed text, the encoded features will be "India, yale, friend". It can be observed that exact English words and the affixes are extracted which will otherwise have a different meaning in Malayalam.

- **Syntactic n-grams:** Inspired by (Sidorov et al., 2013) instead of traditional n-grams, syntactic n-grams (sn-grams) are used as features. sn-grams are constructed by following paths in syntactic trees that enables n-grams to bring syntactic knowledge into ML methods. The difference between sn-grams and traditional n-grams comes in the way of following syntactic relations in syntactic trees by neighbors, whereas traditional n-grams concepts is based on the words presence in a text and are taken from surface strings (Posadas-Durán et al., 2015). Bi and Tri sn-grams are extracted from texts using SNgramExtractor[6] library and the generated sn-grams are attached to feature set. For example, for the sentence 'Economic news have little effect on financial markets' in English, the sn-grams are:

  Bi-sn-grams: news_Economic, have_news, effect_little, have_effect, effect_on, markets_financial, on_markets, have_.

  Tri-sn-grams: effect_on_markets, on_markets_financial.

The combination of above mentioned features are transformed to vectors using CountVectorizer library. Graphical representation of feature engineering module for SACo-Ensemble and SACo-Keras is given in Figure 1.

3.2 Learning Approaches

The proposed models are described below:

3.2.1 SACo-Ensemble:

Three sklearn classifiers, namely, Multi-Layer Perceptron (MLP), eXtreme Gradient Boosting (XGB) and Logistic Regression (LR) are ensembled based on hard majority voting as shown in Figure 2. This model has been trained on count vectors of extracted feature set that consists of char sequences, BPEmb subWords and Syntactic n-grams.

[4] https://scikit-learn.org/stable/

[5] https://pypi.org/project/emoji/

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

Details about the three sklearn classifiers used are given below:

- MLP is a feed-forward Artificial Neural Network (ANN) that consists of at least three layers: an input layer, a hidden layer and an output layer and it is based on a supervised learning technique called back propagation for training. An MLP also can be considered as one of the most traditional types of DL architectures where every element of a previous layer is connected to every element of the next layer. In proposed model, the parameters for MLP, namely, hidden layer sizes, maximum iteration, activation, solver and random state have been set to (150, 100, 50), 300, Relu, Adam and 1 respectively.

- XGB which uses a gradient boosting framework is a decision-tree-based ML algorithm (Chen and Guestrin, 2016) designed to be highly efficient, flexible and portable. It combines hundreds of simple trees to build a more accurate model in such way that in every iteration a new tree for the model will be generated (Zhang and Zhan, 2017). Execution speed and model performance are the two main advantages of XGB classifier[7]. In this study, following configuration has been used for XGB classifier: max_depth is set to 20, and n_estimators, learning rate, colsample_bytree, gamma, reg_alpha, and objective are set to 80, 0.1, 0.7, 0.01, 4, 'multi:softmax' respectively.

- LR is a method primarily used for binary classification problems. However, in the multi-class case, the one-vs-rest (OvR) scheme will be used in training the model. LR classifier has been used with default parameters.

3.2.2 SACo-Keras:

A simple architecture of Keras[8] sequential model has been used to build a Neural Network (NN). Similar to SACo-Ensemble model, feature vectors obtained from feature engineering module are used to train a Keras dense neural network architecture available at

https://www.kaggle.com/ismu94/tf-idf-deep-neural-net

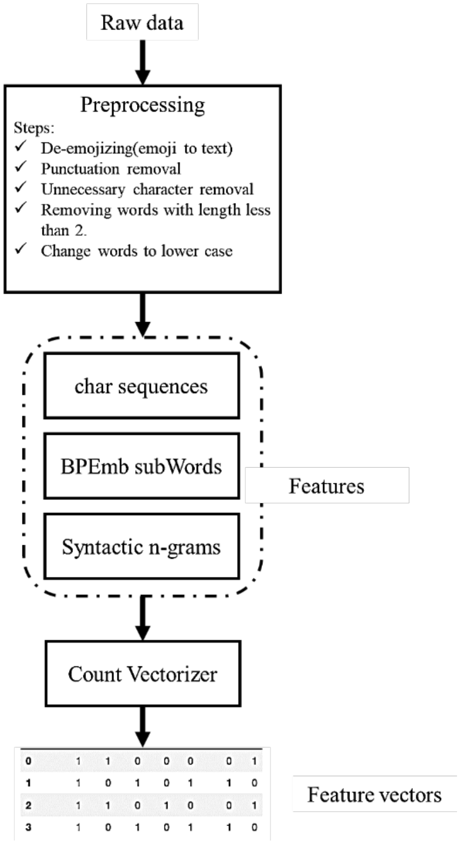

Figure 1: Feature engineering module for SACo-Ensemble and SACo-Keras

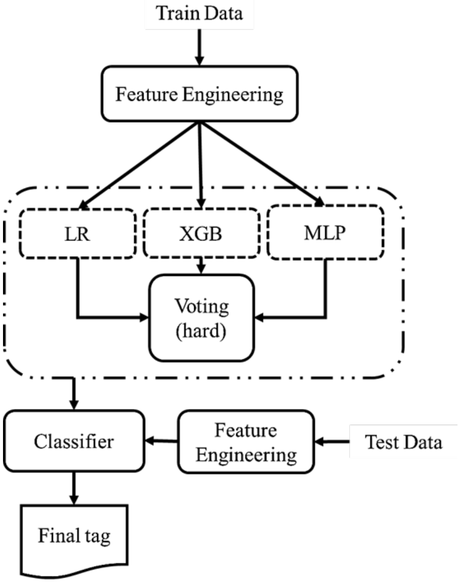

Figure 2: Architecture of SACo-Ensemble model

[7] https://machinelearningmastery.com/gentle-introduction-xgboost-applied-machine-learning

[8] https://keras.io/

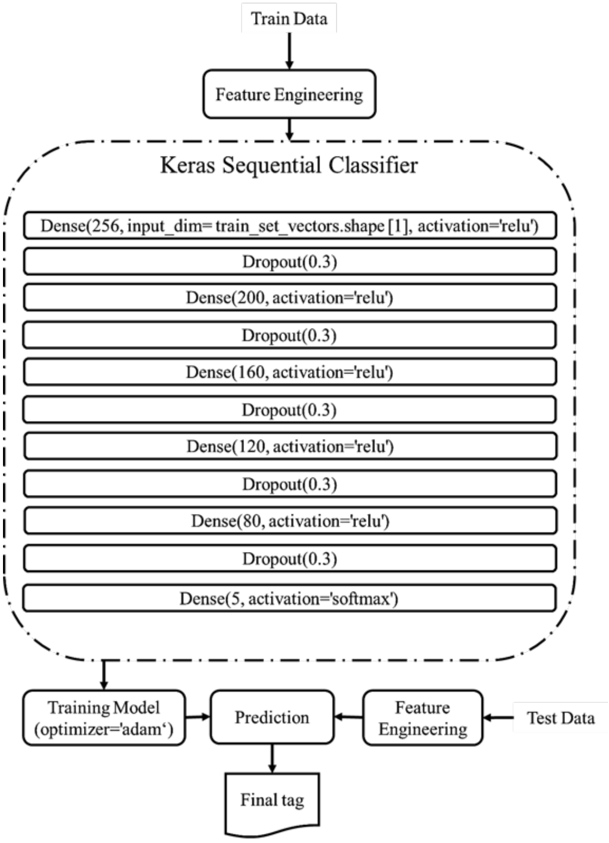
Train Data
Feature Engineering
Keras Sequential Classifier
Dense(256, input_dim= train_set_vectors.shape [1], activation='relu')
Dropout(0.3)
Dense(200, activation='relu')
Dropout(0.3)
Dense(160, activation='relu')
Dropout(0.3)
Dense(120, activation='relu')
Dropout(0.3)
Dense(80, activation='relu')
Dropout(0.3)
Dense(5, activation='softmax')
Training Model (optimizer='adam')
Prediction
Feature Engineering
Test Data
Final tag

Figure 3: Overview of NN layers in SACo-Keras Model

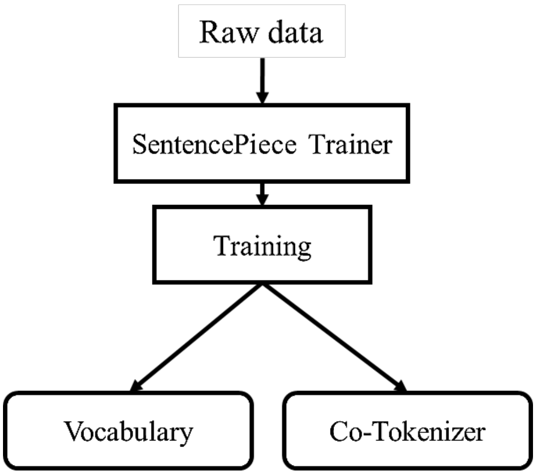

Figure 4: Steps for training CoTokenizer

SACo-Keras model has been trained for 40 epochs with batch size of 128. Figure 3 gives the details of SACo-Keras model with all NN layers and configurations.

3.2.3 SACo-ULMFiT:

This model is based on TL approach where the knowledge obtained from source model (a universal language model in this case) will be transferred to a target model (SA model) affecting the performance of target model (Balouchzahi and Shashirekha, 2020b) (S. Faltl, 2019). ULMFiT model architecture inspired by (Howard and Ruder, 2018) consists of training Language Model (LM) that is a probability distribution over word sequences in a language and then transferring the obtained weights and fine tuning them using texts from training set for SA model.

Fine tuning is the procedure of producing weights for words that are present in training set but are missing in LM due to difference in domains of texts used in training LM and training set for the given task. As these missing words usually are important features, fine tuning the knowledge obtained from pre-trained LM enhances the performance of model.

Training SACo-ULMFiT model includes three major steps, namely, (i) training code-mixed tokenizer and universal language model from raw texts, (ii) transferring the obtained language model and knowledge to final SA model and fine tuning the language model using training set and (iii) training the final SA model for predicting the labels of test set.

The raw text from Dakshina[9] dataset (Roark et al., 2020) along with Ta-En and Ma-En code-mixed datasets (Chakravarthi et al., 2020b) (Chakravarthi et al., 2020a) are used to train code-mixed tokenizer called as Co-Tokenizer (called Co-Tokenizer because it learns from code-mixed raw data and will tokenize texts based on their code-mixing form) which is then used for training code-mixed language models and SA models for Ta-En and Ma-En tasks. The steps for training tokenizer are shown in Figure 4 and the SACo-ULMFiT model architecture for SA of code mixed texts is shown in Figure 5. The text.models tools from Fastai[10] library are used to build both LM and SA models. This tool implements an encoder for an ASGD Weight-Dropped LSTM (AWD-LSTM)

[9] https://github.com/google-research-datasets/dakshina

[10] https://nlp.fast.ai/

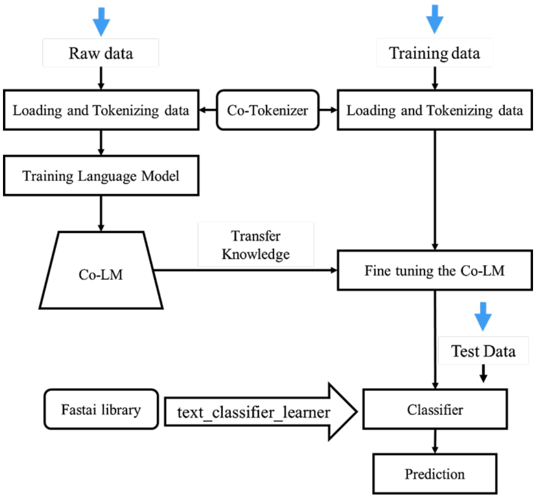

Figure 5: SACo-ULMFiT model architecture

that consists of a word embedding of size 400, 3 hidden layers and 1150 hidden activations per layer that is plugged in with a decoder and classifying layers to create a text classifier (Merity et al., 2017). The preprocessing steps used in this model are the same as that of earlier models.

4 Experimental Results

4.1 Datasets

Datasets used in this study includes unannotated texts from Dakshina dataset which is a collection of text in both Roman and native scripts for 12 South Asian languages such as Malayalam, Tamil, Kannada, etc. (Roark et al., 2020) and labeled datasets for code-mixed SA task in Ma-En and Ta-En (Chakravarthi et al., 2020b) (Chakravarthi et al., 2020a) distributed into 5 classes namely, Positive, Negative, Mixed-Feelings, unknown_state, and other_languages. Since training an efficient LM requires a large amount of data, both Romanized Malayalam data (Ma-Co raw) from Dakshina dataset and the above mentioned code-mixed Ma-En datasets are combined for Malayalam Code-mixing LM (Co-LM). Similarly, Romanized Tamil data (Ta-Co raw) from Dakshina dataset is combined with the above mentioned code-mixed Ta-En dataset for training Tamil Co-LM. Details of the datasets used in this work are given in Table 1 and statistics of the labeled datasets is given in Table 2.

Dataset	Type	No. Sentences
TaCo raw	unannotated	41454
MaCo raw	unannotated	16739
Malayalam-English	annotated	6739
Tamil-English	annotated	15744

Table 1: Datasets used in this work

Labels	Malayalam-English		Tamil-English	
	Training	Test	Training	Test
Positive	2246	565	8484	2075
Negative	600	138	1613	424
Mixed-Feelings	333	70	1424	377
Unknown-state	1505	398	677	173
Other-languages	707	177	397	100

Table 2: Statistics of labeled datasets

Statistics of the datasets given in Table 1[11] illustrate that Ta-En texts are more than Ma-En texts in both annotated and unannotated case. Further, since unannotated data are used in training Co-LMs, it is expected that less number of unannotated data will affect the efficiency of obtained LM and knowledge which in turn will affect the performance of target SA in SACo-ULMFiT model.

The distribution of labels in annotated datasets given in Figure 6 illustrates that both the labeled datasets are imbalanced. However, as the percentage of imbalance sounds to be less in Ma-En dataset, it is expected that proposed model performs better for this dataset compared to Ta-En dataset.

4.1.1 Results

The proposed models are compared with the top 2 ranked models namely, SRJ (Sun and Zhou, 2020) and YNU (Ou and Li, 2020) along with our model SACo-HVC (4th rank in Tamil and 6th rank in

Model	Mal-En			Ta-En		
	P	R	F1	P	R	F1
SRJ	0.74	0.75	0.74	0.64	0.67	0.65
YNU	0.74	0.74	0.74	0.61	0.67	0.63
SACo-HVC	0.68	0.68	0.68	0.60	0.66	0.62
SACo-Ensemble	0.72	0.72	0.72	0.60	0.66	0.62
SACo-Keras	0.70	0.70	0.70	0.61	0.65	0.62
SACo-ULMFiT	0.65	0.66	0.65	0.52	0.62	0.60

Table 3: Results of the SA models

[11] **TaCo**: texts from combination of Ta-En with Romanized Tamil (Dakshina) datasets
MaCo: texts from combination of Ma-En with Romanized Malayalam (Dakshina) datasets

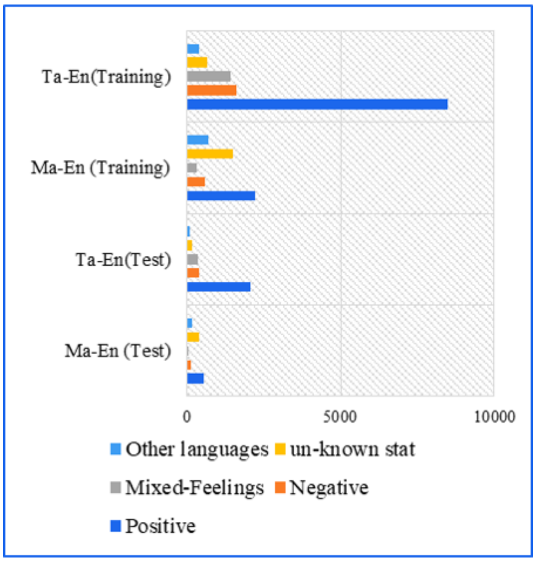

Figure 6: Label distribution in annotated datasets

Malayalam) (Balouchzahi and Shashirekha, 2020a) submitted to "Sentiment Analysis of Dravidian Languages in Code-Mixed Text [12]" shared task in FIRE 2020 and described in (Chakravarthi et al., 2020c). Results obtained in terms of weighted Precision, Recall, and F1-score using Sklearn.metrics library are shown in Table 3. As the results of the proposed models illustrate reasonable performance compared with the top 2 ranked models they still need to be improved.

Comparison of the proposed models and SACo-HVC in terms of weighted F1-score is shown in Figure 7. It can be observed that there is no much improvement in results obtained by new models compared to SACo-HVC for Ta-En dataset. However, SACo-Ensemble and SACo-Keras outperformed the previous SACo-HVC model for Ma-En dataset as the percentage of imbalance is less compared to Ta-En dataset. SACo-ULMFiT model was expected to outperform for Ta-En dataset compared to Ma-En due to large dataset in training LM step, but results show that SACo-ULMFiT for Ma-En dataset obtained better results. Figure 8 presents the history of training each Co-LM for 100 epochs. It illustrates that small number of raw texts for Ma-En Co-LM resulted in less efficient LM model compared to that of Ta-En Co-LM.

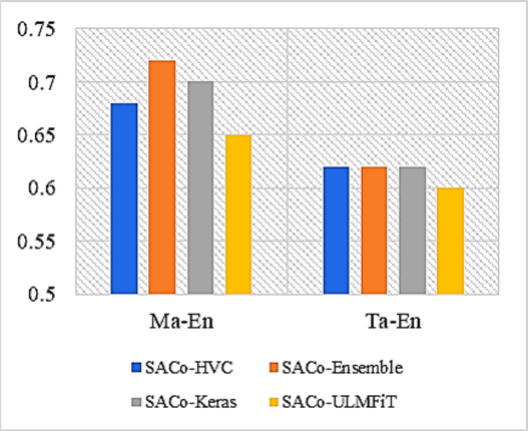

Figure 7: Comparison of our models

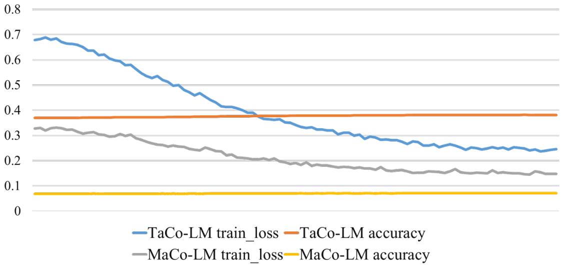

Figure 8: History of training Co-LMs: X and Y axes represent epochs and training accuracy respectively

5 Conclusion and Future Work

This paper describes the three proposed models, namely, SACo-Ensemble, SACo-Keras, and SACo-ULMFiT based on ML, DL and TL for the task of SA for Ta-En and Ma-En code-mixed texts. Ta-En and Ma-En datasets are used to train and evaluate the proposed models. Further, Romanized Tamil and Malayalam unannotated datasets from Dakshina dataset are used to train the code-mixed LMs. The results obtained using proposed models are compared with our previous model, SACo-HVC and with the top 2 ranked models of "Sentiment Analysis for Dravidian Languages in Code-Mixed Text" shared task in FIRE 2020.

The results illustrate that SACo-Ensemble and SACo-Keras models have shown reasonable performance while SACo-ULMFiT exhibited an average performance. Analysis of the results shows that SACo-Ensemble model based on ML approach using a feature set of char sequences, BPEmb subWords, and Syntactic n-grams outperformed the other two proposed models for both datasets. However, the comparison of the proposed models with top 2 ranked models illustrates only a reasonable

[12] https://dravidian-codemix.github.io/2020/index.html

performance. A DL based SACo-Keras model fed with the same feature set as that of SACo-Ensemble model achieved results with very less difference compared to SACo-Ensemble model. This illustrates that there is no much difference between the performances of ML and DL approaches. The SACo-ULMFiT model based TL approach did not perform well which could be due to insufficient unannotated texts for training LMs. As future work, we will collect more raw texts from YouTube and Twitter to build a rich code mixed LM that hopefully will improve the performance of SACo-ULMFiT. Further, we plan to explore different feature sets and feature selection models to improve the performance of our proposed models.

References

Mohammed Arshad Ansari and Sharvari Govilkar. 2018. Sentiment analysis of mixed code for the transliterated Hindi and Marathi texts. *International Journal on Natural Language Computing (IJNLC) Vol*, 7.

F. Balouchzahi and H.L. Shashirekha. 2020a. MUCS@Dravidian-CodeMix-FIRE2020: SACO-Sentiments Analysis for CodeMix Text. In *Forum for Information Retrieval Evaluation*. CEUR Workshop Proceedings.

Fazlourrahman Balouchzahi and HL Shashirekha. 2020b. Puner-parsi ulmfit for named-entity recognition in persian texts. Technical report, EasyChair.

Bharathi Raja Chakravarthi. 2020a. HopeEDI: A multilingual hope speech detection dataset for equality, diversity, and inclusion. In *Proceedings of the Third Workshop on Computational Modeling of People's Opinions, Personality, and Emotion's in Social Media*, pages 41–53, Barcelona, Spain (Online). Association for Computational Linguistics.

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

Bharathi Raja Chakravarthi, Mihael Arcan, and John P. McCrae. 2018. Improving wordnets for under-resourced languages using machine translation. In *Proceedings of the 9th Global Wordnet Conference*, pages 77–86, Nanyang Technological University (NTU), Singapore. Global Wordnet Association.

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 and Vigneshwaran Muralidaran. 2021. Findings of the shared task on Hope Speech Detection for Equality, Diversity, and Inclusion. In *Proceedings of the First Workshop on Language Technology for Equality, Diversity and Inclusion*. Association for Computational Linguistics.

Bharathi Raja Chakravarthi, Vigneshwaran Muralidaran, Ruba Priyadharshini, and John Philip McCrae. 2020b. 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. 2020c. 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.

Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794.

Nurendra Choudhary, Rajat Singh, Ishita Bindlish, and Manish Shrivastava. 2018. Sentiment analysis of code-mixed languages leveraging resource rich languages. *arXiv preprint arXiv:1804.00806*.

Nikhil Kumar Ghanghor, Parameswari Krishnamurthy, Sajeetha Thavareesan, Ruba Priyadharshini, and Bharathi Raja Chakravarthi. 2021a. IIITK@DravidianLangTech-EACL2021: Offensive Language Identification and Meme Classification in Tamil, Malayalam and Kannada. In *Proceedings of the First Workshop on Speech and Language Technologies for Dravidian Languages*, Online. Association for Computational Linguistics.

Nikhil Kumar Ghanghor, Rahul Ponnusamy, Prasanna Kumar Kumaresan, Ruba Priyadharshini, Sajeetha Thavareesan, and Bharathi Raja Chakravarthi. 2021b. IIITK@LT-EDI-EACL2021:

Hope Speech Detection for Equality, Diversity, and Inclusion in Tamil, Malayalam and English. In *Proceedings of the First Workshop on Language Technology for Equality, Diversity and Inclusion*, Online.

Siddhanth U Hegde, Adeep Hande, Ruba Priyadharshini, Sajeetha Thavareesan, and Bharathi Raja Chakravarthi. 2021. UVCE-IIITT@DravidianLangTech-EACL2021: Tamil Troll Meme Classification: You need to Pay more Attention. In *Proceedings of the First Workshop on Speech and Language Technologies for Dravidian Languages*. Association for Computational Linguistics.

Benjamin Heinzerling and Michael Strube. 2017. Bpemb: Tokenization-free pre-trained subword embeddings in 275 languages. *arXiv preprint arXiv:1710.02187*.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*.

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.

Yash Kumar Lal, Vaibhav Kumar, Mrinal Dhar, Manish Shrivastava, and Philipp Koehn. 2019. De-mixing sentiment from code-mixed text. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 371–377.

Stephen Merity, Nitish Shirish Keskar, and Richard Socher. 2017. Regularizing and optimizing LSTM language models. *arXiv preprint arXiv:1708.02182*.

X. Ou and H. Li. 2020. YNU@Dravidian-CodeMix-FIRE2020: XLM-RoBERTa for Multi-language Sentiment Analysis. In *Forum for Information Retrieval Evaluation*. CEUR Workshop Proceedings.

Juan-Pablo Posadas-Durán, Ilia Markov, Helena Gómez-Adorno, Grigori Sidorov, Ildar Batyrshin, Alexander Gelbukh, and Obdulia Pichardo-Lagunas. 2015. Syntactic n-grams as features for the author profiling task. *Working Notes Papers of the CLEF*.

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.

Karthik Puranik, Adeep Hande, Ruba Priyadharshini, Sajeetha Thavareesan, and Bharathi Raja Chakravarthi. 2021. IIITT@LT-EDI-EACL2021-Hope Speech Detection: There is always hope in Transformers. In *Proceedings of the First Workshop on Language Technology for Equality, Diversity and Inclusion*. Association for Computational Linguistics.

Brian Roark, Lawrence Wolf-Sonkin, Christo Kirov, Sabrina J Mielke, Cibu Johny, Isin Demirsahin, and Keith Hall. 2020. Processing south asian languages written in the latin script: the dakshina dataset. *arXiv preprint arXiv:2007.01176*.

C. Hackober S. Faltl, M. Schimpke. 2019. *ULMFiT: State-of-the-art in text analysis*. https://humboldt-wi.github.io/blog/research/information_systems_1819/group4_ulmfit/.

Grigori Sidorov, Francisco Velasquez, Efstathios Stamatatos, Alexander Gelbukh, and Liliana Chanona-Hernández. 2013. Syntactic dependency-based n-grams: More evidence of usefulness in classification. In *International Conference on Intelligent Text Processing and Computational Linguistics*, pages 13–24. Springer.

R. Sun and X. Zhou. 2020. SRJ @ Dravidian-CodeMix-FIRE2020:Automatic Classification and Identification Sentiment in Code-mixed Text. In *Forum for Information Retrieval Evaluation*. CEUR Workshop Proceedings.

Shardul Suryawanshi and Bharathi Raja Chakravarthi. 2021. 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.

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.

Konthala Yasaswini, Karthik Puranik, Adeep Hande, Ruba Priyadharshini, Sajeetha Thavareesan, and Bharathi Raja Chakravarthi. 2021. IIITT@DravidianLangTech-EACL2021: Transfer Learning for Offensive Language Detection in Dravidian Languages. In *Proceedings of the First Workshop on Speech and Language Technologies for Dravidian Languages*. Association for Computational Linguistics.

Licheng Zhang and Cheng Zhan. 2017. Machine learning in rock facies classification: an application of xgboost. In *International Geophysical Conference, Qingdao, China, 17-20 April 2017*, pages 1371–1374. Society of Exploration Geophysicists and Chinese Petroleum Society.