Amrita@LT-EDI-EACL2021: Hope Speech Detection on Multilingual Text

Thara S, Ravi Teja T, Sai Rahul K
Amrita Vishwa Vidyapeetham,
Amritapuri, Kerala, India
thara@am.amrita.edu,ravitejatasubilli@gmail.com,
kothamasusairahul@gmail.com

Abstract

Analysis and deciphering code-mixed data is imperative in academia and industry, in a multilingual country like India, in order to solve problems apropos Natural Language Processing. This paper proposes a bidirectional long short-term memory (BiLSTM) with attention-based approach, in solving hope speech detection problem. Using this approach an F1-score of 0.73 (9 th rank) in Malayalam-English data set was achieved from a total of 31 teams who participated in the competition.

1 Introduction

Code-mixing may be defined as the phenomenon of mixing of elements of speech or text, from two or more languages, into a single utterance. Code-mixing is widely observed in social social-media platforms (Chakravarthi, 2020b). Code mixing imposes diverse challenges in Natural Language Processing (NLP) tasks like sentiment analysis (SA), named entity recognition (NER), and parts-of-speech (POS) tagging (Jose et al., 2020; Priyadharshini et al., 2020).

Nowadays, as more people lean on online communication platforms, it is incumbent on bloggers to stop the spread of negativity through online posts (Mandl et al., 2020; Chakravarthi et al., 2020, 2021). Studies have been carried out to surveil and cease the consumption of abusive comments on social-media platforms like You-tube and Twitter (Ghanghor et al., 2021a; Yasaswini et al., 2021; Hegde et al., 2021). Besides curbing dissemination of negativity, it is imperative for users online to endorse and promote positivity and hope (Chakravarthi and Muralidaran, 2021). As studies have shown that hope is an important factor in prevention of self-immolation and self-harm Snyder et al. (2002). Hope is generally attributed as assurance, encouragement, support, or motivation instilled in individuals by their peers or surroundings Chakravarthi (2020a); Puranik et al. (2021); Ghanghor et al. (2021b)

However, there is not much study done in understanding hope speech in the domain of code mixed communications. Hope Speech Detection for Equality, Diversity, and Inclusion-EACL 2021 is first such task [1] to detect hope speech in low resourced code-mixed languages like Malayalam-English, Tamil-English, Kannada-English.

2 Literature Survey

Since Hope Speech Detection for Equality, Diversity, and Inclusion-EACL 2021 is the first task in Hope Speech Detection, there is not much focus in this area. Most of the studies have been done in the NLP aspects of NER, SA and POS tagging on code mixed data.

In Sravani et al. (2018) the authors experimented with various word embedding techniques using 6 different classifiers and showed that Term Frequency and Inverse Document Frequency (TF-IDF) word embeddings yielded better results for NER task on English-Hindi code-mixed data.

In Thara and Poornachandran (2018) the authors discusses on the literature survey of code-mixed problems and datasets.

In Sasidhar et al. (2020) authors created a corpus of 12000 Hindi-English code mixed texts from various online communication platforms and annotated them as Happy, Sad and Anger based on the emotions. The authors achieved an accuracy of 83.21% using CNN-BiLSTM architecture.

In Sreelakshmi et al. (2020) the authors have

[1] https://competitions.codalab.org/competitions/27653

Proceedings of the First Workshop on Language Technology for Equality, Diversity and Inclusion, pages 149–156
April 19, 2021.

used Support Vector Machine (SVM)-Radial Basis Funcrion (RBF) classifier on features extracted using fastText for hate speech detection in Hindi-English code mixed corpus of 10000 texts which are equally divided into hate and non-hate classes. The authors obtained an accuracy of 0.75% using SVM-RBF algorithm.

In Liu et al. (2020) used an adversarial training with a pre-trained multilingual model XLM-R Conneau et al. (2020) on code-mixed Hindi-English data [2] and achieved an F1-score of 0.75 (1st rank) for SA.

3 Proposed Method

3.1 Data Description

Dataset consists of comments taken from YouTube[3] video comment section. Data set is classified into three classes namely Hope speech, Non-hope speech, not-Malayalam.

Hope speech: Comments having positive intentions, infused with inspiration provided by peers to an individual, promoting wellbeing, joy and happiness, comments promoting morals like equality, diversity and inclusion are labelled as Hope speech.

Non-hope speech:Comments that are biased and criticize without considering the outcome, and promote cultural, racially hatred interactions are classified as non-hope speech

Not-Malaylam: Comments that are not Malayalam or in Malayalam-English are categorized as not-Malayalam.

We merged the train and development data sets into a single data set. The merged data set is comprised of 9634 comments in total (Table 1), whereas the test data set (Table 2) has an aggregate of 1071 comments. 90 % of this merged data as were used as train data (Table 3) and 10 % as development data set (Table 4). Majority of the comments in the data set are labelled as non-hope speech, the data set as such is imbalanced

Class	Count	Percentage
Not-Malayalam	786	8.2 %
Hope speech	1860	19.3 %
Non-hope speech	6988	72.5 %
Total	9634	

Table 1: Total Data Statistics.

Class	Count	Percentage
Not-Malayalam	101	9.4 %
Hope speech	194	18.1 %
Non-hope speech	776	72.5 %
Total	1071	

Table 2: Test Data Statistics

Class	Count	Percentage
Not Malayalam	707	8.2 %
Hope speech	1674	19.3 %
Non-hope speech	6289	72.5 %
Total	8670	

Table 3: Train Data Statistics

Class	Count	Percentage
Not Malayalam	79	8.2 %
Hope speech	186	19.3 %
Non-hope speech	699	72.5 %
Total	964	

Table 4: Development Data Statistics

[2] https://competitions.codalab.org/competitions/20654

[3] https://www.youtube.com/

3.2 Data Preprocessing

Manual Cleaning: Data set contains 109159 comments/posts from YouTube video comment Section is comprised of abbreviations like: RSS (Rashtriya Swayamsevak Sangh), CAA (Citizenship Amendment Act), UAE (United Arab Emirates), ICMR (Indian Council of Medical Research), and usage of common internet slang like bro, sis, subs (subscriber) is widely observed throughout the data set. We manually expanded most of the abbreviation in train, development, and test data. We replaced commonly used internet slang like u (you), luv (love), vid (video), bro (brother), sis (sister), btw (by the way), rip (rest in peace) with the complete form of the word. We manually corrected misspelled English words. Numerical short-cuts like "3.5k" , "4 M" were rewritten in complete numerical form i.e. as "3500", "4000000". We desegmented words which have English prefix and Malayalam suffix like "machineil", "storyil", "collegeil" as "machine il", "story il","college il". Dates which are in numerical format like "29-12-2020" are rewritten as "December 29 2020". Symbols "&","+","%" are replaced with "and", "plus", "percentage".

Preprocessing: After manually cleaning the data, the following steps were used to preprocess the data.

Step 1: Remove emojis in the data and convert all words in Latin script into lower case

Step 2: Remove punctuation marks, @ from user mentions, # from hashtags.

Step 3: Replace characters and words which are occurring consecutively more than two times with a single occurrence.

3.3 Feature Extractions

Word2Vec and FastText algorithms were used for feature extractions

Word2Vec: Word2Vec (Mikolov et al., 2013) is a neural network based model which takes in text corpus as inputs, and yields word embeddings as output (a set of vectors). It is done using two different methods, one is Continuous-Bag-of Words (CBOW) and the other one is Skip-gram

In CBOW, initially every word in the vocabulary is given a random vector representation and the model is given a set of neighboring words, both from front and back of the word. The number of neighboring words is a hyperparameter, called as window size. The model is then asked to predict the present word. When training completes, the model outputs unique vector representation to each word in the vocabulary.

In Skip-gram method, just like with CBOW the model starts with a random vector representation for each word in the corpus. The model is then given a word and is asked to predict its surrounding neighbors, from the front and back of the word.

FastText : FastText, an extended version of Word2Vec is based on (Joulin et al., 2016; Bojanowski et al., 2017). FastText uses sub- word information to extract features. FastText first generates character n-grams for each word.
Example: 3-gram for word “bharat" would be < bh, har,ara, rat, at >. Unlike Word2Vec, FastText can handle Out-of-Vocabulary (OOV) words making use of the n-gram approach. OOV word vector representations can be obtained by adding the vector representations of sub words. Gensim implementation[4] were used for both FastText and Word2Vec.

3.4 Classification Model:

CNN: Studies have shown that CNN can be used in NLP. In (Kim, 2014), author shows that CNN with Word2Vec can attain remarkable results in sentence classification, with minimal hyperparameter training. The author has used seven different sets of data for this study. In (Zhang and Wallace, 2016) an extensive study on sensitivity of the model’s performance for sentence classification was performed, based on model architecture and other hyperparameters.

In our experiments, we implemented a single layer 1-dimensional convolutional neural network (1D-CNN) with 256 kernels of size 1, followed by two feed-forward layers, where the first neural network has 256 units and the last one has 3 units, which is acting as the classification layer. A dropout is (Srivastava et al., 2014)used for regularization. The key-idea of dropout is to prevent units in the neural network (NN) from co-adapting too much, by randomly dropping units and their connection in the network during training.

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

Bi-LSTM: Long Short Term Memory networks(LSTM) (Hochreiter and Schmidhuber, 1997) are variants of RNN, designed to overcome the long term dependency problems in RNN. LSTMs address the long-term dependency problem by using an internal memory called cell state. LSTMs have the privilege to remove or add new information to this cell state by regulating a structure called gates.

We have implemented Bidirectional LSTM (BiLSTM) with attention mechanism. Bi-LSTM gives up representation of the input text from both directions, using two different LSTMs. Embeddings from Word2Vec and FastText are fed into the BiLSTM, the hidden states of BiLSTM layer are then fed into the attention layer which then gives a weighted context vector. The context vector is conveyed to a feed-forward layer having 3 units to classify the input.

4 Experiments and Results

We implemented all our models in tensorflow[5].

4.1 Experimental Methods

Word Embeddings: We experimented with embeddings of 100, 200, and 300 dimensions. We used Word2Vec and FastText (both Skip-gram and CBOW) embedding techniques, which yielded a best weighted F1-score.

Dropout and L2 regularization: We have considered a range of values between 0.0 and 1.0 for both dropout and regularization.

Optimizers: Adagrad, Adadelta, Adam, RMSprop, Stochastic Gradient Descent (SGD) were used as optimizers; the optimizer which resulted in a better F1-score was selected as the preferred optimizer.

Learning rate: We have considered a range of values between 10^{-5} to $10^{-}3$.

Epochs: Range of values between 30 and 100 with an interval of 10 were considered.

Data imbalance: The majority of the comments in the dataset belong to non-hope speech category. To address this data imbalance we experimented with up-sampling and data augmentation using back translation. We applied random up-sampling method to minority classes.

[5] https://www.tensorflow.org

4.2 Results

CNN with Word2Vec embeddings: Word embeddings from Word2Vec are feed into 1-D CNN; the results were evaluated using weighted F1-score. Various hyperparameters and word embeddings dimensions were experimented with. 1-D CNN with up-sampled data using Word2Vec embeddings (CNN_W2V_UP) obtained a weighted F1-score of 0.73 on development data set. 1-D CNN without any augmentation technique (CNN_W2V_NORM) obtained a weighted F1-score of 0.62 on development data set. 1-D CNN with data augmentation using back translation (CNN_W2V_AUG) obtained a weighted F1-score of 0.69 on development data set. See Table 5 and Figures: 2,3,4

Model	Dev_data	Test Data
CNN_W2V_NORM	0.62	0.62
CNN_W2V_AUG	0.69	0.68
CNN_W2V_UP	**0.73**	**0.75**

Table 5: **F1-score for development and test data**.CNN_W2V_UP: 1-D CNN with up-sampled data using Word2Vec embeddings, CNN_W2V_NORM: 1-D CNN without any augmentation technique using Word2Vec embeddings, CNN_W2V_AUG: 1-D CNN with data augmentation using back translation with Word2Vec embeddings.

CNN with FastText embeddings: Word embeddings from FastText are feed into 1-D CNN and we evaluated the results using weighted F1-score. We experimented with various hyperparameters and word embeddings dimensions. 1-D CNN with up-sampled data using FastText embeddings (CNN_ft_UP) obtained a weighted F1-score of 0.74 on development data set. 1-D CNN without any augmentation technique (CNN_ft_NORM) obtained a weighted F1-score of 0.70 on development data set. 1- D CNN with data augmentation using back translation (CNN_ft_AUG) obtained a weighted F1-score of 0.63 on development data set, see Table 6 and Figures: 5,6,7

Bi-LSTM with attention using FastText: Our initial experiments have shown that up-sampling yields better results compared to data augmentation using back translation, so we have used FastText as the feature extraction method and we used random up-sampling with attention based Bi-LSTM model. We obtained a weighted F1-score of 0.85 on the

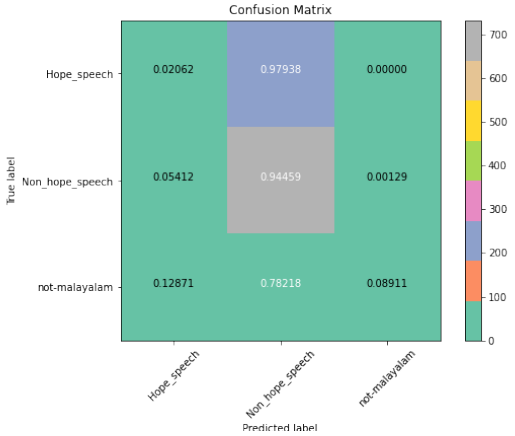

Figure 1: Confusion matrix for test data: 1-D CNN without any augmentation technique using Word2Vec embeddings.

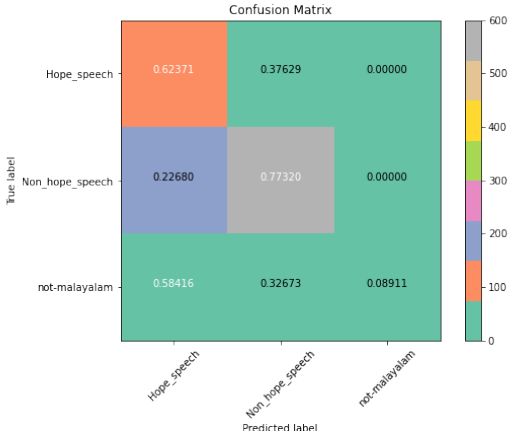

Figure 2: Confusion matrix for test data: 1-D CNN with data augmentation using back translation with Word2Vec embeddings.

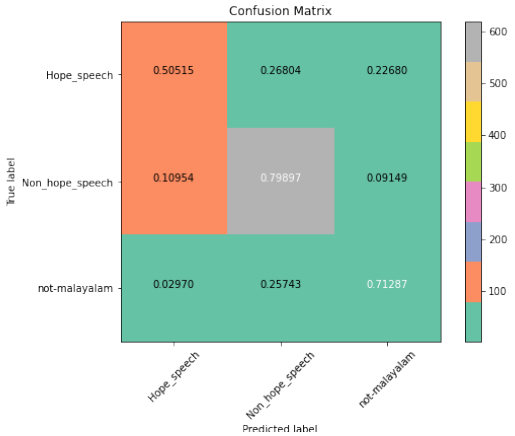

Figure 3: Confusion matrix for test data: 1-D CNN with up-sampled data using Word2Vec embeddings

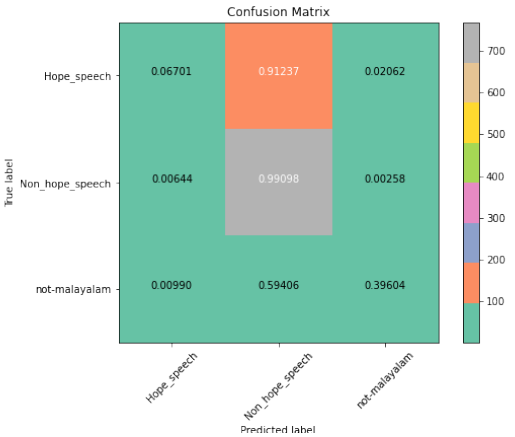

Figure 4: Confusion matrix for test data: 1-D CNN without any augmentation technique using FastText embeddings.

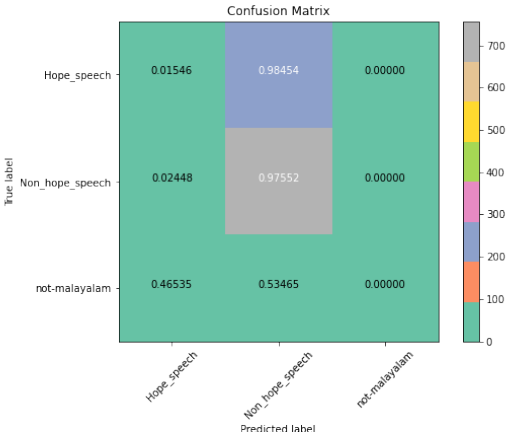

Figure 5: Confusion matrix for test data: 1-D CNN with data augmentation using back translation with FastText embeddings.

Model	Dev_data	Test Data
CNN_ft_AUG	0.63	0.62
CNN_ft_NORM	0.70	0.70
CNN_ft_UP	**0.74**	**0.76**

Table 6: **F1-score for development and test data**.CNN_ft_UP: 1-D CNN with up-sampled data using FastText embeddings, CNN_ft_NORM: 1-D CNN without any augmentation technique using FastText embeddings, CNN_ft_AUG: 1-D CNN with data augmentation using back translation with FastText embeddings.

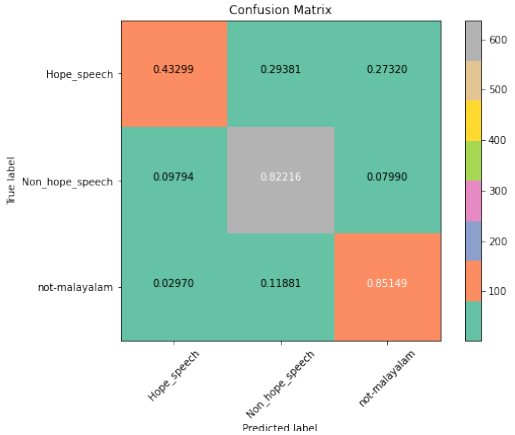

Figure 6: Confusion matrix for test data: 1-D CNN with up-sampled data using FastText embeddings.

development data set. We achieved an F1-score of 0.73 in the test data. As the F1-score for development data set of this model was the best compared to the rest of the models we submitted this model's results in the task. On the test data this model obtained an F1-score of 0.73, see Figure 8.

Model	Dev_data	Test Data
BiLSTM_ft_UP	0.85	0.73
CNN_W2V_UP	**0.73**	**0.75**
CNN_ft_UP	**0.74**	**0.76**

Table 7: **F1 score for development and test data**.CNN_W2V_UP: 1-D CNN with up-sampled data using Word2Vec embeddings,CNN_ft_UP: 1-D CNN without any augmentation technique using Word2Vec embeddings, Bi-LSTM_ft_UP: Bi-LSTM with up sampled data using FastText embeddings.

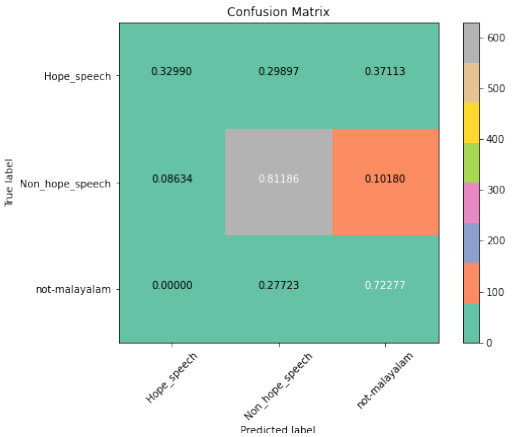

Figure 7: Confusion matrix for test data: Bi-LSTM with up-sampled data using FastText embeddings.

5 Conclusion

We evaluated the performance of our models on the test data using sklearn F1-score [6], we found that 1-D CNN on up-sampled data with both Word2Vec and FastText embeddings performed better than the Bi-LSTM model which we submitted for the task. However, Bi-LSTM performed relatively better on the development data set (see Table 7). We also observed that random up-sampling yielded better results compared to data augmentation with back translation method.

References

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information.

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, M Anand Kumar, John Philip McCrae, Premjith B, Soman KP, and Thomas Mandl. 2020. Overview of the track on HASOC-Offensive Language Identification-DravidianCodeMix. In *Working Notes of the Forum for Information Retrieval Evaluation (FIRE 2020).*

[6] https://pypi.org/project/scikit-learn/

CEUR Workshop Proceedings. In: CEUR-WS. org, Hyderabad, India.

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, 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.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale.

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.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

N. Jose, B. R. Chakravarthi, S. Suryawanshi, E. Sherly, and J. 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.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. Bag of tricks for efficient text classification.

Yoon Kim. 2014. Convolutional neural networks for sentence classification.

Jiaxiang Liu, Xuyi Chen, Shikun Feng, Shuohuan Wang, Xuan Ouyang, Yu Sun, Zhengjie Huang, and Weiyue Su. 2020. Kk2018 at SemEval-2020 task 9: Adversarial training for code-mixing sentiment classification. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 817–823, Barcelona (online). International Committee for Computational Linguistics.

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.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space.

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.

Turaga Sasidhar, Premjith B., and Soman Kp. 2020. Emotion detection in hinglish(hindi+english) code-mixed social media text. *Procedia Computer Science*, 171:1346–1352.

C. Snyder, Hal Shorey, Jennifer Cheavens, Kim Pulvers, Virgil III, and Cynthia Wiklund. 2002. Hope and academic success in college. *Journal of Educational Psychology*, 94:820–826.

Lolla Sravani, Atla Reddy, and Thara Sasidharan. 2018. A comparison study of word embedding for detecting named entities of code-mixed data in indian language. pages 2375–2381.

K. Sreelakshmi, Premjith B., and Soman Kp. 2020. Detection of hate speech text in hindi-english code-mixed data. *Procedia Computer Science*, 171:737–744.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958.

S Thara and Prabaharan Poornachandran. 2018. Code-mixing: A brief survey. In *2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pages 2382–2388. IEEE.

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.

Ye Zhang and Byron Wallace. 2016. A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification.