Detecting Abusive Comments in Multiple Languages: A Multilingual Approach

Sai Krishna Sangeetha, Venkata Sai Kumar Ganesula
Department of Computer Science
University of Central Florida
Orlando, Florida, United States
saikrishna.sangeetha@knights.ucf.edu, venkatasaikumarg@knights.ucf.edu

ABSTRACT

India is a diverse country with a rich linguistic heritage. There were 122 major languages and 1599 other languages officially spoken in India, according to 2011 Census of India[9]. The precise number can vary depending on the standards used to define a distinct language, but it is important to keep in mind that some of these "languages" may be dialects or variations of the same language. Social media usage by people around the globe is rapidly increasing as the internet access is expanding. As a result, there is more offensive, abusive, and hateful content on social media. For applications like controversial event extraction, creating chatterbots, content recommendation, and sentiment analysis, hate speech detection on the social media platforms like Twitter, Facebook, TikTok etc., is essential[2]. Since India has abundant languages and more than a million people can speak at least a language, we would like to experiment and identify abusive comments on the Indian languages. The ability to categorize a tweet or comment as racist, sexist, or neither is how we define this task. This task is very difficult because natural language constructs are so complex. Our project aims to utilize natural language processing techniques to identify negative comments posted on social media platforms. Once we have identified these negative comments, we can focus on promoting positive content on these platforms. This approach will help us create a more positive and constructive online community. We implement several baseline models for the classification on the dataset. We have found that our proposed model, MuRIL outperforms other models by interpreting the semantic expressions better.

KEYWORDS

Hate Speech Detection, Naïve Bayes, BERT, MuRIL, LSTM

1 Introduction

Hate speech is a pervasive problem on social media platforms, and its impact can be felt in societies worldwide[6]. While India has the second-largest population globally, it is also home to a diverse linguistic landscape with numerous dialects and languages. Despite the prevalence of hate speech in India, existing multilingual models have been ineffective in detecting abusive content in Indian languages. Furthermore, hate speech and abusive language online not only have a negative impact on individuals but can also lead to societal harms such as discrimination and violence. In India, the issue of hate speech is particularly complex due to the country's diversity of languages and cultures[5]. It is crucial to develop effective techniques to detect and prevent the spread of hate speech on social media platforms to promote a safe and inclusive online environment. This study aims to address this issue by exploring the performance of various machine learning and deep learning techniques for detecting abusive comments in Indian languages, and proposing a novel approach utilizing MuRIL, a state-of-the-art multilingual model pre-trained on 17 Indian languages[2]. The findings of this study can have implications for the development of effective and inclusive social media policies and strategies.

2 Problem Statement

The objective of this project is to create a model that can determine whether a comment is abusive or non-abusive. The dataset is made up of comments written in different Indian dialects. For the best accuracy on the dataset, we want to investigate and contrast various machine learning and deep learning techniques in this project and compare with the proposed model.

3 Related Work

Hate speech is a growing concern in the digital age and detecting it on social media platforms has become a critical task. Many hate speech detection systems are designed for English, and there is a lack of work done in Indian languages. The earlier effort in developing resources for the hate speech detection was mainly focused on English language (Waseem and Hovy, 2016; Davidson et al., 2017; Founta et al., 2018). Recently, in an effort to create multilingual hate speech datasets, several shared task competitions have been organized (HASOC (Mandl et al., 2019), OffensEval (Zampieri et al., 2019),, TRAC (Kumar et al., 2020), etc.), and multiple datasets such as Hindi (Modha et al., 2021), Danish (Sigurbergsson and Derczynski, 2020), Greek (Pitenis et al., 2020), Turkish (Çöltekin, 2020), Mexican Spanish (Aragón et al., 2019), etc. have been made public. Initial approaches to detecting abusive speech involved using lexicons, handcrafted features, and metadata.

Pinkesh Badjatiya, Shashank Gupta, Manish Gupta, Vasudeva Varma et al. “Deep Learning for Hate Speech Detection in Tweets” in this research paper they proposed a deep learning approach to detect hate speech in tweets. The authors use a convolutional neural network and a long short-term memory network to classify the tweets. The results show that the deep learning approach outperforms traditional machine learning methods and can be used to build automated systems for hate speech detection and prevention. The authors suggest future work should focus on addressing challenges in detecting hate speech such as sarcasm and context to improve the performance of the model.

However, recent advancements in natural language processing have led to the development of transformer-based models, which have shown state-of-the-art performance for hate speech detection tasks. Multilingual models like mBERT and XLM-R have been proposed to address semantic understanding across multiple languages, particularly in resource-poor settings. For Indic languages, MuRIL and IndicBERT have been developed. MuRIL has been trained on 17 Indic languages and English language datasets using MLM and TLM techniques, while IndicBERT is a multilingual ALBERT model trained on 12 Indian languages. “MuRIL: Multilingual Representations for Indian Languages.” (Khanuja et al., 2021) aims to enrich reciprocity from one language to another. This model uses a BERT base architecture pre-trained from scratch using the Common Crawl, Wikipedia, PMINDIA, and Dakshina corpora for 17 Indian languages and their transliterated counterparts. These models provide a promising solution for hate speech detection in Indian languages and can aid in combating online hate speech.

4 Dataset

The dataset for this task is curated for an challenge hosted by IEEE BigMM. The dataset consists of abusive and non-abusive comments which were posted on Moj app in 13 languages accompanied by contextual user data (https://www.kaggle.com/c/iiitd-abuse-detection-challenge/data). The dataset contains 352,386 non-abusive datapoints and 312,656 abusive data points. Dataset has 8 features, namely, language, post_index, commentText, report_count_comment, report_count_post, like_count_comment, like_count_post, label. We have focussed on the commentText which has comment as string and label which is used to predict, 0 is non-abusive and 1 is abusive.

5 Data Analysis

We have split the dataset into two parts, the first part will have 598,537 datapoints for training and validation, the later will have 66,505 data points which is considered as unseen data and fed to the proposed model to validate the test accuracy.

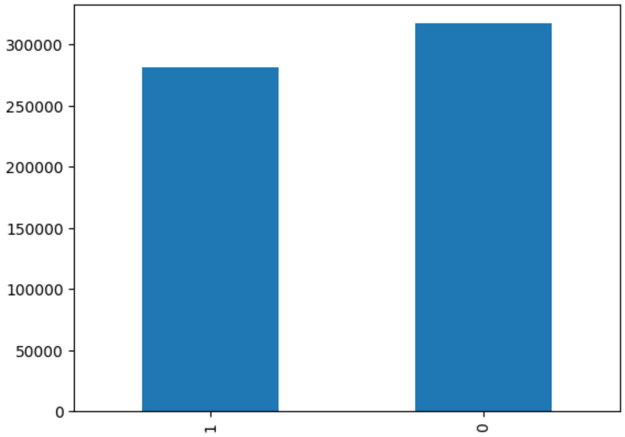


Figure 1: Data Distribution by Class

Above plot illustrates the data points constitute for the data distribution by class. Where, non-abusive comments are represented by ‘0’ and abusive comments are represented by ‘1’. We can observe that the dataset has balanced classes.

Now, we plot the number of datapoints available for each language in the dataset and we observe that Hindi has the highest number of datapoints and Telugu has the second highest and there are some languages that have very few datapoints compared to these languages.

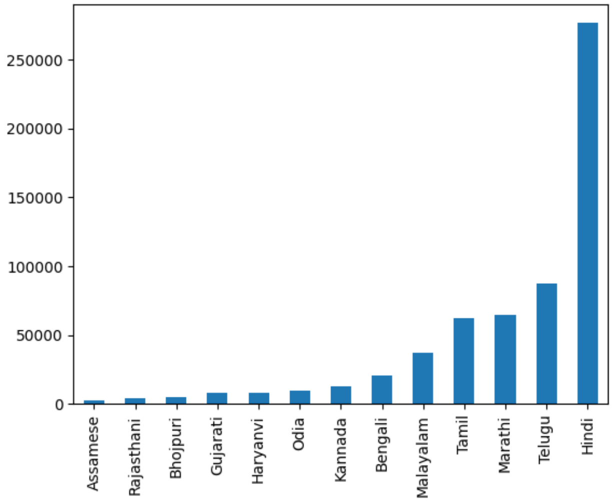


Figure 2: Data Distribution by Language

We have split the dataset as 90% for training and 10% for validation.

6 Methodology

6.1 Baseline Model 1

In our project, we are tokenizing comments in Indian languages, and to accomplish this task, we have selected the Indic tokenizer.

The reason for this choice is that traditional tokenizers available in Python libraries such as spaCy and NLTK may not be well-suited to handle Indian languages. The Indic NLP library, which was specifically designed for Indian languages, provides us with a solution to this problem. The Indic tokenizer works by identifying punctuation boundaries within the text and using them to generate tokens[4].

To generate word vectors from the tokenized data in our project, we used the Tf-idf Vectorizer. We opted for this approach because pre-trained embeddings like gloVe embeddings and word2vec are designed primarily for English text and may not be effective for Indian languages. The Tf-idf Vectorizer was selected because it calculates the significance of each word in a document based on its frequency within that document, compared to its_frequency in the corpus. This method enables us to assess the originality of each word and generate appropriate word vectors.

In preparation for fitting the Logistic Regression model to our data, we begin by computing the ratio, using Bayes' rule[18].

$$\frac{P\left(y = \frac{1}{x}\right)}{P\left(y = \frac{0}{x}\right)}$$

This ratio is then multiplied by the word vectors to gain insight into the likelihood of a given datapoint generating a label of 1 or 0. Specifically, if the ratio is greater than 1, the model is more likely to predict a label of 1, whereas if the ratio is less than 1, the model is more likely to predict a label of 0. This approach allows us to make more informed predictions and improve the accuracy of our model. We can see the architecture for model 1 in Figure 3.

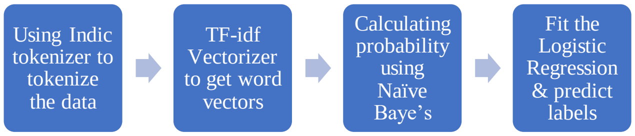

Using Indic tokenizer to tokenize the data
TF-idf Vectorizer to get word vectors
Calculating probability using Naïve Baye's
Fit the Logistic Regression & predict labels

Figure 3: Baseline Model 1 Architecture

6.2 Baseline Model 2

Word embeddings have been shown to be an effective method for achieving high accuracy in classification tasks. In our approach, we trained an embeddings layer on the training dataset to enable prediction on the testing dataset. By using this method, we can improve the accuracy of our model and make more accurate predictions on previously unseen data.

6.2.1 Baseline Model 2a

To prepare the data for embedding generation, we first removed punctuation and tokenized the text. We then built a vocabulary based on the training set to enable uniform input size for the embedding layer. The embeddings were then passed through a fully connected neural network for prediction.

For training the model, we used the RMSprop optimizer and experimented with varying numbers of epochs to achieve the best accuracy on the test set. Our hyperparameters included a vocabulary size of 10000, feature dimension of 8, max length of 128, and batch size of 512. By fine-tuning these parameters, we were able to optimize the model's performance and generate accurate predictions. We can observe the architecture for model 2a in Figure 4.

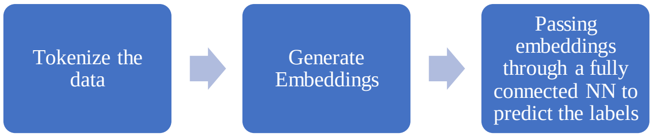

Tokenize the data
Generate Embeddings
Passing embeddings through a fully connected NN to predict the labels

Figure 4: Baseline Model 2a Architecture

6.2.2 Baseline Model 2b

This model builds upon Model 2a and makes some key modifications to improve its performance. While GloVe embeddings are commonly used in NLP tasks for English language data, they are not suitable for Indian languages, so we needed to take a different approach.

To replicate the output of GloVe embeddings, we increased the output dimension of the embeddings layer from 8 to 100. This change allowed the model to learn more effectively and led to an increase in accuracy.

Our hyperparameters for this model included a vocabulary size of 10000, number of epochs ranging from 10 to 30, feature dimension of 100, max length of 128, and batch size of 512. By fine-tuning these parameters, we were able to improve the model's performance and generate more accurate predictions. We can observe the architecture for model 2b in Figure 5.

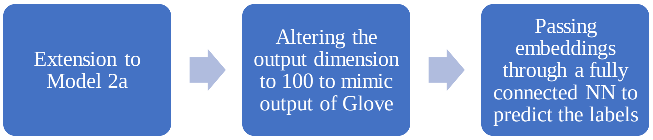

Extension to Model 2a
Altering the output dimension to 100 to mimic output of Glove
Passing embeddings through a fully connected NN to predict the labels

Figure 5: Baseline Model 2b Architecture

6.2.3 Baseline Model 2c

This model builds upon Model 2a and incorporates more advanced deep learning techniques such as Bidirectional LSTM and Batch normalization.

Bidirectional LSTM is a sequence-based model that works well with textual data in classification tasks. By incorporating this approach, we were able to create a more complex model that achieved higher accuracy than Model 2a and similar accuracy to Model 2b.

The model was configured and fine-tuned in a similar manner to Model 2a. Our hyperparameters included a vocabulary size of 10000, number of epochs ranging from 10 to 20, feature dimension of 8, max length of 128, and batch size of 512. By adjusting these parameters, we were able to optimize the model's performance and generate more accurate predictions.

Different epoch values were tried to achieve the best accuracy for the models. After 10 epochs, both Model 2a and Model 2c began to overfit, while Model 2b showed signs of overfitting after just 3 epochs. We can observe the architecture for model 2c in Figure 6.

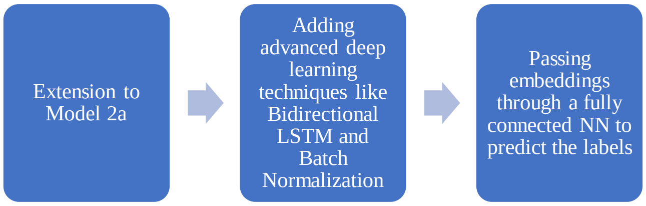

Extension to Model 2a
Adding advanced deep learning techniques like Bidirectional LSTM and Batch Normalization
Passing embeddings through a fully connected NN to predict the labels

Figure 6: Baseline Model 2c Architecture

7 Proposed Model

To generate word embeddings for the comments data, MURIL was used in this study. MURIL was chosen because it is a BERT model pre-trained on a vast amount of data from Indian languages, which could generate highly meaningful embeddings for this case. The study employed a fully connected neural network to predict labels based on the embeddings generated[2].

For the pre-processing, before using MURIL, the data was tokenized into a specific format using AutoTokenizer. Then we saved the pre-trained tokenizer for the "MURIL" language model. We encoded a list of texts using the pre-trained tokenizer. Once we get the encodings, we used TesorFlow function to prepare the training data for the language model.

We first defined two input layers using the Input() function from Keras, one for the input IDs and one for the attention mask. The shapes of these layers are (config.max_len,), which is likely the maximum length of the input sequences i.e., 64, and their data types are both tf.int32. The names of these layers are 'input_ids' and 'attention_mask', respectively.

The transformer_model is then applied to the input layers to generate the transformer output tensor. Since this is a classification task, only the first token of the transformer output tensor is used, which represents the CLS token.

The transformer output tensor is then passed through a batch normalization layer and a dropout layer with a rate of 0.1. This is followed by a 1D convolutional layer with a kernel size of 1, which helps to reduce the number of features in the tensor. The resulting tensor is flattened using the Flatten() function, passed through another batch normalization layer, and then passed through a dense layer with a sigmoid activation function, which outputs a probability score for the binary classification task.

Finally, a Keras Model is created using the input layers and the output layer, and compiled the model with the Adam optimizer, binary cross-entropy loss, and accuracy metric. With the generated embeddings we will predict the labels. The architecture is illustrated in Figure 7.

The hyperparameters for the model are, epochs = 12, learning_rate = 1e-5, batch_size = 64, dropout = 0.1.

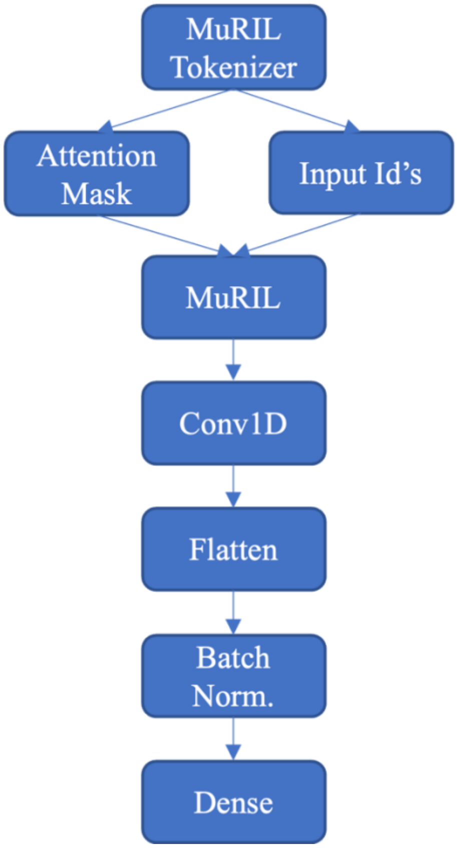
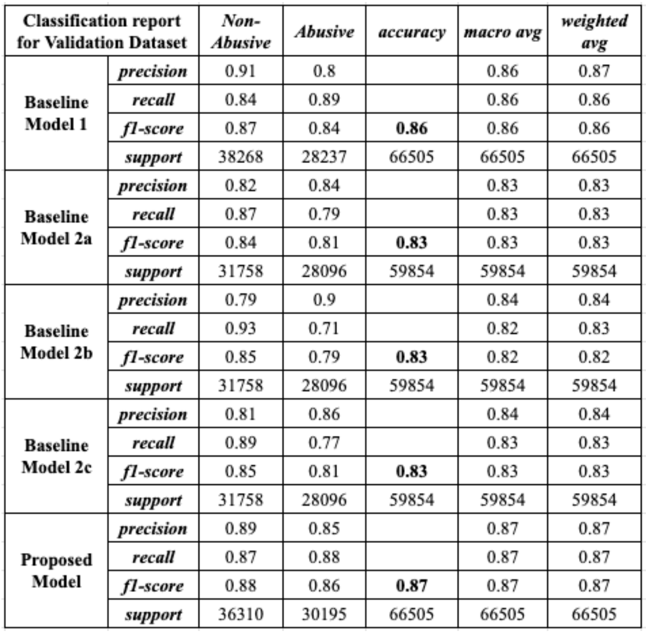


Figure 7: Proposed Model Architecture

8 Results

We measure the performance of our models in terms of accuracy, precision, recall, and f1-score to stay consistent with the body of existing literature. Together, these metrics ought to be able to assess the classification system's effectiveness at differentiating between the two classes, such as abusive and non-abusive.

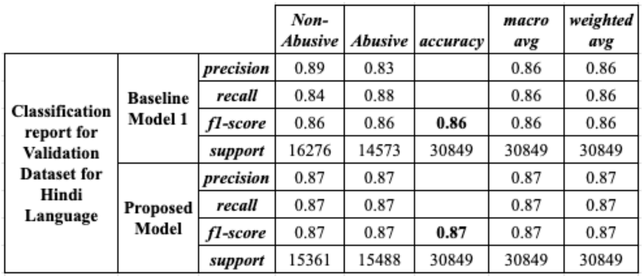
Our proposed model has performed well on the dataset, we have achieved a fairly good validation score when compared to the baseline models. We can observe that from the classification report for validation dataset, Results 1, Baseline model 1 has achieved an accuracy of 0.86, and baseline models 2a, 2b, 2c have achieved an accuracy of 0.83, whereas the proposed model has achieved an accuracy of 0.87. Clearly, we can say that the proposed model has outperformed the baseline models.



Classification report for Validation Dataset		Non-Abusive	Abusive	accuracy	macro avg	weighted avg
Baseline Model 1	precision	0.91	0.8		0.86	0.87
	recall	0.84	0.89		0.86	0.86
	f1-score	0.87	0.84	**0.86**	0.86	0.86
	support	38268	28237	66505	66505	66505
Baseline Model 2a	precision	0.82	0.84		0.83	0.83
	recall	0.87	0.79		0.83	0.83
	f1-score	0.84	0.81	**0.83**	0.83	0.83
	support	31758	28096	59854	59854	59854
Baseline Model 2b	precision	0.79	0.9		0.84	0.84
	recall	0.93	0.71		0.82	0.83
	f1-score	0.85	0.79	**0.83**	0.82	0.82
	support	31758	28096	59854	59854	59854
Baseline Model 2c	precision	0.81	0.86		0.84	0.84
	recall	0.89	0.77		0.83	0.83
	f1-score	0.85	0.81	**0.83**	0.83	0.83
	support	31758	28096	59854	59854	59854
Proposed Model	precision	0.89	0.85		0.87	0.87
	recall	0.87	0.88		0.87	0.87
	f1-score	0.88	0.86	**0.87**	0.87	0.87
	support	36310	30195	66505	66505	66505

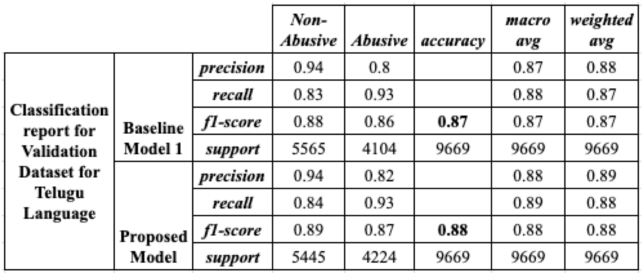
Result 1: Classification Report for Validation Dataset for all Models

Validation evaluation metrics were computed for Hindi and Telugu, which have the highest number of data points, as well as for Assamese and Rajasthani, which have the lowest number of data points. We noticed that the model performed equally well in both cases, indicating that it can learn effectively by identifying similarities such as phonology and script used in Indian languages. From the below results, we can see that the accuracies in all conditions are favorable to the proposed model.



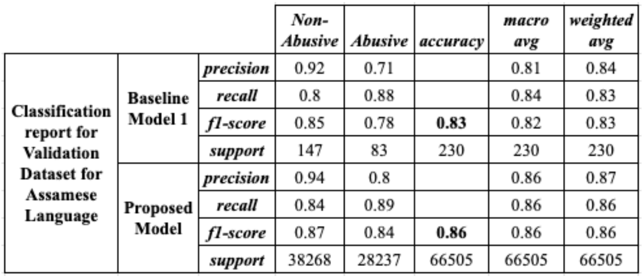
			Non-Abusive	Abusive	accuracy	macro avg	weighted avg
Classification report for Validation Dataset for Hindi Language	Baseline Model 1	precision	0.89	0.83		0.86	0.86
		recall	0.84	0.88		0.86	0.86
		f1-score	0.86	0.86	**0.86**	0.86	0.86
		support	16276	14573	30849	30849	30849
	Proposed Model	precision	0.87	0.87		0.87	0.87
		recall	0.87	0.87		0.87	0.87
		f1-score	0.87	0.87	**0.87**	0.87	0.87
		support	15361	15488	30849	30849	30849

Result 2: Classification Report for Validation Dataset for Hindi Language



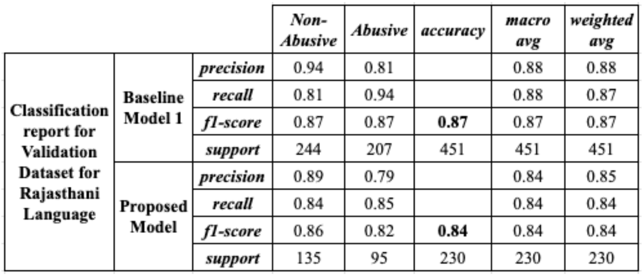
			Non-Abusive	Abusive	accuracy	macro avg	weighted avg
Classification report for Validation Dataset for Telugu Language	Baseline Model 1	precision	0.94	0.8		0.87	0.88
		recall	0.83	0.93		0.88	0.87
		f1-score	0.88	0.86	**0.87**	0.87	0.87
		support	5565	4104	9669	9669	9669
	Proposed Model	precision	0.94	0.82		0.88	0.89
		recall	0.84	0.93		0.89	0.88
		f1-score	0.89	0.87	**0.88**	0.88	0.88
		support	5445	4224	9669	9669	9669

Result 3: Classification Report for Validation Dataset for Telugu Language



			Non-Abusive	Abusive	accuracy	macro avg	weighted avg
Classification report for Validation Dataset for Assamese Language	Baseline Model 1	precision	0.92	0.71		0.81	0.84
		recall	0.8	0.88		0.84	0.83
		f1-score	0.85	0.78	**0.83**	0.82	0.83
		support	147	83	230	230	230
	Proposed Model	precision	0.94	0.8		0.86	0.87
		recall	0.84	0.89		0.86	0.86
		f1-score	0.87	0.84	**0.86**	0.86	0.86
		support	38268	28237	66505	66505	66505

Result 4: Classification Report for Validation Dataset for Assamese Language



			Non-Abusive	Abusive	accuracy	macro avg	weighted avg
Classification report for Validation Dataset for Rajasthani Language	Baseline Model 1	precision	0.94	0.81		0.88	0.88
		recall	0.81	0.94		0.88	0.87
		f1-score	0.87	0.87	**0.87**	0.87	0.87
		support	244	207	451	451	451
	Proposed Model	precision	0.89	0.79		0.84	0.85
		recall	0.84	0.85		0.84	0.84
		f1-score	0.86	0.82	**0.84**	0.84	0.84
		support	135	95	230	230	230

Result 5: Classification Report for Validation Dataset for Rajasthani Language

9 Hypothesis

As per the evaluations and results, our proposed model has outperformed the traditional machine learning and deep learning methods. From this we can draw an hypothesis that, will the proposed model have capability to be trained on a subset of languages and can be utilized for identifying whether a comment is abusive or non-abusive in other languages? From this hypothesis, we can have three conditions,

1. Training model on any two languages (say, Hindi and Tamil), then predicting on dataset pertaining to other languages like Telugu or Bengali.
2. Training model on all available languages then predicting on the dataset pertaining to one selected language (say Malayalam).
3. Training model on the data points from selected language (say Malayalam) then predicting on the dataset pertaining to the selected language (Malayalam).

Hypothesis 1

The model can be trained on a small number of languages and used to determine whether a comment is abusive or not in languages other than English. The model could, for instance, be trained using data from the Hindi and Tamil training sets while making predictions using data from the Telugu or Bengali training sets.

Reasoning

The embeddings for many languages will have the same dimensions because we intend to employ pre-trained models. As a result, the model trained on a dataset for one language can be used to infer on a dataset for another language because the model's parameters are based on the feature vector of the input data point.

Hypothesis 2

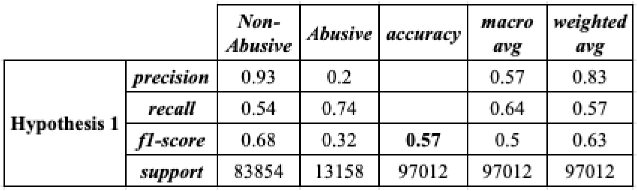
Training on all the languages available in the dataset will give better performance than training the model for single language.

Reasoning

As the data for a single language is limited, and the model might not generalize well for lesser amount of data; We believe that model will perform better when exposed to higher amount of data and even though the additional data is from a different language.

10 Hypothesis Testing & Results

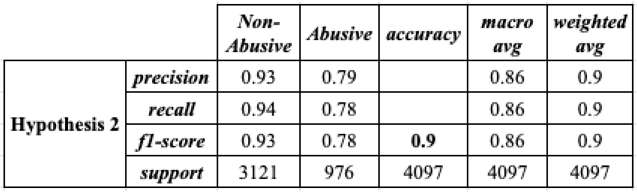
Inference 1: To evaluate the hypothesis 1, we have trained the MURIL based model with training dataset pertaining to Hindi and Tamil language. The trained model was then evaluated on dataset with Telugu language. We were able to perform prediction on dataset pertaining to Telugu.



		Non-Abusive	***Abusive***	***accuracy***	***macro avg***	***weighted avg***
Hypothesis 1	***precision***	0.93	0.2		0.57	0.83
	recall	0.54	0.74		0.64	0.57
	f1-score	0.68	0.32	**0.57**	0.5	0.63
	support	83854	13158	97012	97012	97012

Result 6: Classification report for Telugu language from the model trained using Hindi and Tamil

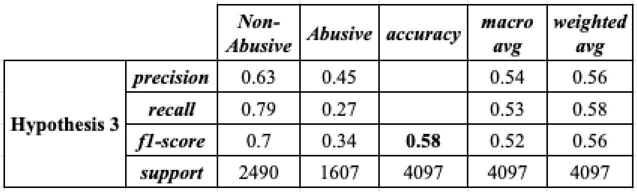
Inference 2: To evaluate the hypothesis 2, we have trained the MURIL based model with the complete training dataset and a validation set with data points from Malayalam language. The trained model was then evaluated on the validation set.



		Non-Abusive	***Abusive***	***accuracy***	***macro avg***	***weighted avg***
Hypothesis 2	***precision***	0.93	0.79		0.86	0.9
	recall	0.94	0.78		0.86	0.9
	f1-score	0.93	0.78	**0.9**	0.86	0.9
	support	3121	976	4097	4097	4097

Result 7: Classification report for Malayalam after training on all languages

Inference 3: Another model was trained with only data points from Malayalam language. This trained model was then evaluated on the validation set.



		Non-Abusive	***Abusive***	***accuracy***	***macro avg***	***weighted avg***
Hypothesis 3	***precision***	0.63	0.45		0.54	0.56
	recall	0.79	0.27		0.53	0.58
	f1-score	0.7	0.34	**0.58**	0.52	0.56
	support	2490	1607	4097	4097	4097

Result 8: Classification report for Malayalam after training on Malayalam language

From the above evaluations, we can say that although training of first model was performed not only on a particular language but also from other languages in the dataset, the model was able to learn better than that of the model which was trained only on that particular language.

11 Conclusion

This paper proposes MuRIL model for detecting abusive comments. We compared tradition machine learning and deep learning methods with the proposed MuRIL model. From the above experiments, evaluations and hypothesis testing, we can clearly state that the proposed model has outperformed the traditional machine learning and deep learning techniques. Based on the hypothesis evaluations, it can be concluded that the first model, which was trained on multiple languages in the dataset, outperformed the model trained on a single language. This indicates that the embeddings from different languages are beneficial for the model, and once trained, the model can be used for inference on any language. This is particularly advantageous when data for a specific language is scarce but abundant data is available for other languages.

12 Software Specifications

Python, Jupyter, Google Colab, TensorFlow, Keras, PyTorch, Scikit-Learn, NLTK, Hugging Face, iNLTK, StanfordNLP, Indic NLP, Transformers etc.,

REFERENCES

[1] Pinkesh Badjatiya, Shashank Gupta, Manish Gupta, Vasudeva Varma et al. "Deep Learning for Hate Speech Detection in Tweets."

[2] Simran Khanuja, Diksha Bansal, Sarvesh Mehtani, Savya Khosla, Atreyee Dey, Balaji Gopalan, Dilip Kumar Margam, Pooja Aggarwal, Rajiv Teja Nagipogu, Shachi Dave, et al. 2021. Muril: Multilingual representations for indian languages. arXiv preprint arXiv:2103.10730.

[3] "Detecting Hate Speech in Social Media Using Deep Learning Techniques" by N. Pandya, D. Bhatt, and D. Doshi.

[4] IndicBERT: A Multilingual Language Model for Indian Languages" by Divyanshu Kakwani, Himanshu Sharma, Prakhar Gupta, Abhishek Kumar, and Manish Shrivastava.

[5] Divyanshu Kakwani, Anoop Kunchukuttan, Satish Golla, Gokul N.C., Avik Bhattacharyya, Mitesh M. Khapra, and Pratyush Kumar. 2020. "IndicNLPSuite: Monolingual Corpora, Evaluation Benchmarks and Pre-trained Multilingual Language Models for Indian Languages. In Findings of EMNLP"

[6] Mithun Das, Binny Mathew, Punyajoy Saha, Pawan Goyal, and Animesh Mukherjee. 2020. Hate speech in online social media. ACM SIGWEB Newsletter, (Autumn):1–8.

[7] Binny Mathew, Ritam Dutt, Pawan Goyal, and Animesh Mukherjee. 2019. Spread of hate speech in online social media. In Proceedings of WebSci. ACM.

[8] Mikel Artetxe, Sebastian Ruder, and Dani Yogatama. 2020. On the cross-lingual transferability of monolingual representations. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 4623–4637, Online. Association for Computational Linguistics.

[9] INDIA. 2011. Census of india, 2011. https: //www.censusindia.gov.in/2011Census/ C-16_25062018_NEW.pdf.

[10] 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. In Proceedings of the 12th Language Resources and Evaluation Conference, pages 2413–2423, Marseille, France. European Language Resources Association.

[11] Monirah A Al-Ajlan and Mourad Ykhlef. Optimized twitter cyberbullying detection based on deep learning. In 2018 21st Saudi Computer Society National Computer Conference (NCC), pages 1–5. IEEE, 2018.

[12] Valerio Basile, Cristina Bosco, Elisabetta Fersini, Nozza Debora, Viviana Patti, Francisco Manuel Rangel Pardo, Paolo Rosso, Manuela Sanguinetti, et al. Semeval-2019 task 5: Multilingual detection of hate speech against immigrants and women in twitter. In 13th International Workshop on Semantic Evaluation, pages 54–63. Association for Computational Linguistics, 2019.

[13] Steve Durairaj Swamy, Anupam Jamatia, and Björn Gambäck. Studying generalisability across abusive language detection datasets. In Proceedings of the 23rd conference on computational natural language learning (CoNLL), pages 940–950, 2019.

[14] Tharindu Ranasinghe and Marcos Zampieri. An evaluation of multilingual offensive language identification methods for the languages of india. Information, 12(8):306, 2021.

[15] K, K.; Wang, Z.; Mayhew, S.; Roth, D. Cross-Lingual Ability of Multilingual BERT: An Empirical Study. In Proceedings of the International Conference on Learning Representations, Addis Ababa, Ethiopia, 1 May–26 April 2020.

[16] Y. Kim. Convolutional Neural Networks for Sentence Classification. In EMNLP, pages 1746–1751, 2014.

[17] C. Nobata, J. Tetreault, A. Thomas, Y. Mehdad, and Y. Chang. Abusive Language Detection in Online User Content. In WWW, pages 145–153, 2016.

[18] Ng, A.Y. & Jordan, M. I. (2002). On Discriminative vs. Generative Classifiers: A comparison of Logistic Regression and Naive Bayes, Neural Information Processing Systems, Ng, A.Y., and Jordan, M. (2002).