Hostile Language Detection in Hindi Posts using Neural Techniques

Aditi Damle
Electrical and Computer Engineering
Carnegie Mellon University
Pittsburgh, PA 15213
aditimid@andrew.cmu.edu

Shivani Shekhar
Electrical and Computer Engineering
Carnegie Mellon University
Pittsburgh, PA 15213
sshekha2@andrew.cmu.edu

Siddhant Jagtap
Electrical and Computer Engineering
Carnegie Mellon University
Pittsburgh, PA 15213
sjagtap2@andrew.cmu.edu

1 Introduction

With the exponential growth of user generated content on social media, there are more instances of hostile language these days. In the last few years, there has been interesting research in this domain in the form of research papers and online shared tasks. Most of this work has targeted English data but users use other languages as well. With nearly 500M Hindi speakers in India and around the world, it is important to detect hostile Hindi posts on social media. This project is about classifying social media posts in the Indian language, Hindi (Devanagari script). We are using a variety of hostile posts in Hindi collected from Twitter and Facebook. These posts are either hostile or not hostile but classifying a post simply as hostile or not-hostile is not sufficient. To get better classification we try to understand the sentiment of the non hostile post as hateful, fake, defamation or offensive. In this paper we discuss two types of experiments: 1. Coarse-grained task- Here we classify a post as hostile or non-hostile and 2. Fine-grained Task- Here we sub-categorize a post as either hateful, fake, defamation or offensive.

In Section 2, we discuss relevant work on hostile language data and neural architecures for hostile language detection in various languages. In Section 3, we describe the dataset used for this task. In Section 4, we explain our baseline model as well as other models we have implemented. Section 5 describes the experimental setup. In Section 6, we report our results and analyze them in Section 7. In Section 8 we suggest directions for future work and provide ideas for improving our results.

The link to our GitHub repository: `https://github.com/siddjags/nnfornlp-project/tree/main`

2 Related Work

We explored the following literature revolving around hate/hostile language detection in social media posts.

2.1 Hostile Language Data and Labeling

Bhardwaj et al. proposed a new multi-dimensional hostility detection dataset in Hindi language [4], which was part of the CONSTRAINT-2021 shared task on hostile post detection in Hindi. The authors annotated about 8200 posts as hostile or non-hostile across Facebook and Twitter. They also

assigned fine-grained hostile labels to each hostile post, i.e., fake, hate, offensive, and defamation. An interesting observation is that hostile posts have a higher average number of letters per post even if the average number of words in hostile posts is lower than in non-hostile posts. They then bench-marked the dataset using common binary classification techniques and reported the weighted F1 scores. We have chosen this dataset for our project.

[9] and [10] talk about creation of new datasets to counter bias and imbalance prevailing in majority of existing natural language datasets. They have explained in detail the process of dataset creation and have tested them with state-of-the-art algorithms. [10] also shows that models trained with a bias label for hate speech detection, performed better, implying that bias and hate are intertwined.

Davidson et al. demonstrate the shortcomings of lexical detection methods with low precision for hate speech tasks as they fail to distinguish between hate speech and offensive language [6]. They created dataset from tweets but with three labels - hate, offensive and neither. By testing it with common algorithms, they emphasized separation of hate speech and offensive language. We have focused on maintaining this distinction in our implementation as well. In [14], the authors focus on semantics for effective classification. They claim that hateful content exhibits a 'long tail' pattern compared to non-hateful text due to their lack of unique, discriminative linguistic features which make them difficult to classify. This causes current methods to classify most content as non-hate. They proposed CNN based models to classify tweets that lack discriminative features, that performed better than others. However, detecting hate content solely based on linguistic content is still a problem.

2.2 Neural Architectures for Hostile Language Detection

Badjatiya et al. compare various deep learning models for multi-class hate speech detection [3], where a tweet can be classified as either sexist, racist or neither. The paper by Koratana et al. [8] describes toxic speech and the approaches to address challenges involved in using a model at run time to detect hate speech. They used the dataset published by Google Jigsaw on Kaggle, "Toxic Comment Classification Challenge". In [11], the authors propose a CNN classifier that uses GloVe embedding vectors to capture semantic information. The problem statement here is to predict hate speech in a tweet as soon as it is posted by the user. They proposed an end to end CNN system that acts as an n-gram feature extractor depending on the kernel size. Aluru et al. explore various deep learning models that can be used for multilingual hate speech detection [2]. The authors combine various datasets for nine languages. CNN-GRU and BERT-based models are evaluated on this aggregated multilingual dataset. Furthermore, Ghosh Roy et al. propose a novel approach to represent emojis and hashtags present in social media posts [12]. The paper by Abu-Farha et al. presents a multi-task learning framework that involves CNN, max-pooling as well as bi-LSTM layers [1].

For Indic languages (Hindi in particular), Bhatnagar et al. suggest an ensemble-learning based approach for hostile speech detection in Hindi online posts [5]. They combine techniques such as support vector machines with deep learning models to achieve competitive results. In [13], Shekhar et al. use a combination of deep neural networks and XGBoost based models for both coarse-grained as well as fine-grained classification of hostile posts. Finally, Kamal et al. propose a novel architecture for the hostile speech detection task in Hindi [7]. They use the multi-label Hindi language dataset described in [4]. Various neural baseline models such as binary classifiers, multi-label classifiers and multi-task learning models are used by the authors for experimentation. In order to improve upon existing baseline models, the authors propose an approach called Auxiliary Task Based Binary Sub-Classification. A detailed explanation on this approach can be found in Section 4. For all their experiments, the authors use BERT variants such as HindiBERT, Indic BERT and HindiBERTa which have been exclusively trained on Hindi/Indic text.

It is evident from the literature review that research done on hate/hostile speech detection in Hindi language is somewhat limited. This is surprising as Hindi is the third most widely spoken language in the world. Therefore, we aim to contribute to this research area by extending the work done by [7].

3 Dataset

The dataset was obtained from the website of CONSTRAINT 2021 shared task in which the authors of [7] had participated. This dataset consists of 8192 online posts in Hindi (devanagari script) out of which 4358 samples belong to the non-hostile category, while the remaining 3834 posts convey

one or more hostile dimensions and corresponding labels ('offensive', 'hate', 'hostile', 'defamation', 'fake'). There are 1638, 1132, 1071, and 810 posts for fake, hate, offensive, and defamation classes in the annotated dataset, respectively. The dataset is stored in csv format and is already split into train, validation and test sets in the ratio of 70,20,10. It is worth noting that a hostile post can have multiple labels. For example, a post can be labeled as both 'fake' and 'offensive'.

Table 1: CONSTRAINT 2021 Hindi Dataset Summary

	#Fake	#Hate	#Offensive	#Defamation	#Hostile	#Non-hostile
Train	1144	792	742	564	2678	3050
Validation	160	103	110	77	376	435
Test	334	237	219	169	780	873

From Table 1, it is evident that the dataset is imbalanced with respect to most hostile sub-classes. For instance, there are very few samples for some labels like defamation. This inherent imbalance in the dataset can result in unstable model performance. In order to account for this imbalance, we describe certain preprocessing and data augmentation techniques in Section 5.1.

4 Methodology

The objective of this project is to accurately detect if a Hindi post contains hostile language. As mentioned in Section 2.2, Kamal et al. take this task one step further by proposing models that perform multi-label classification of hostile posts into fake, hate, offensive and defame classes [7]. The following subsections describe the baseline model as well as several other architectures we have experimented with.

4.1 Baseline Model

Our baseline model is the best-performing model in [7]. It divides the task into an Auxiliary/course-grained task (hostile/non-hostile) and multiple fine-grained sub-tasks (hostile sub-classes). Therefore, this approach is called Auxiliary Task Based Binary Sub-Classification. The logits generated by the course-grained classifier are concatenated with each of the sub-tasks. The authors claim that the auxiliary classifier helps in capturing information that is common across all sub-classes of hostile posts. Fig 1 illustrates this baseline architecture.

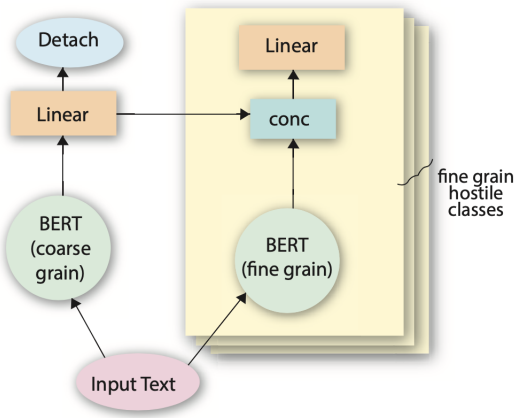

Figure 1: Baseline Model (Adapted from Kamal et al. [7])

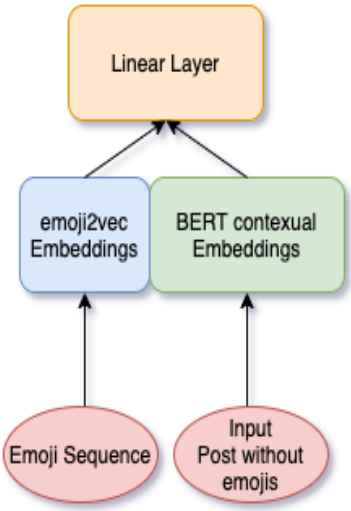

Figure 2: BERT + Emoji

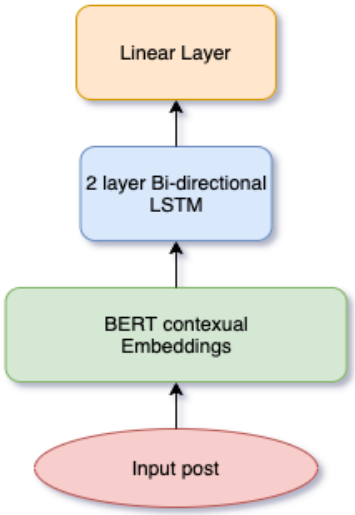

Figure 3: BERT + Bi-LSTM

4.2 BERT Classifier with concatenated Emoji Embeddings (BERT + Emoji)

The architecture of this model is similar to the coarse-grained classifier of the baseline model. However, the BERT contextual embeddings are concatenated with vector representations of the emoji sequence present in the post. In the absence of emojis, a zero vector is used in place of the emoji embeddings. On the other hand, if there are multiple emojis in a post, the average vector representation is used in this case. Fig 2 illustrates this approach. This model was mainly suited for the coarse-grained task.

4.3 Modified Auxiliary Classifier (Baseline + Emoji)

In this approach, the auxiliary classifier in the baseline model is replaced with the architecture described in Section 4.2. This implies that the classification uses a vector representation that comprises of the coarse-grained BERT embeddings, fine-grained BERT embeddings as well as the emoji embeddings. The high-level architecture is identical to Fig 1.

4.4 BERT Classifier with Bi-LSTM Layer (BERT + Bi-LSTM)

This approach involves a bidirectional LSTM layer stacked on top of the BERT embeddings and was mainly used for the fine-grained tasks. The final hidden state of the sequence of tokens is fed as input to the bi-LSTM. Since the LSTM layer is bi-directional, the hidden state corresponding to the [CLS] token is then fed to a linear layer for classification. Fig 3 illustrates this architecture.

4.5 Ensemble Classifier

In order to further improve the performance on the fine-grained tasks, a majority vote classifier was designed using three BERT + Bi-LSTM models. A detailed description of this approach can be found in Section 5.2.

5 Experiments

5.1 Preprocessing

The original dataset consists of raw Facebook/Twitter posts and is annotated using a string of multiple classes/labels. To each post, we assigned binary labels for all tasks, i.e. hostile, offensive, defame, hate, fake, depending on the presence of these class names in the post's label string. This way the

label string is mapped to five different binary label columns for coarse-grained as well as fine-grained tasks.

All posts were cleaned by removing stop words, punctuation and special characters. As we observed better performance in the presence of URLs, we decided to retain them as is. We believe that URLs add semantic value and facilitate the classification tasks. For instance, a hostile post which simply quotes content from a referenced URLs, may be incorrectly classified as hostile in the absence of the URL.

Furthermore, emojis were separated from the original posts and stored as independent emoji sequences. For models requiring emoji embeddings, pre-trained emoji2vec representations were used. For every emoji, a 300-dimensional vector representation was obtained using emoji2vec. Additional details regarding emoji embeddings can be found in Section 4.2.

For all fine-grained tasks, the preprocessed posts were back-translated in order to perform data augmentation. Such data augmentation strategies can be effective when the original dataset is skewed with respect to a particular class/label.

5.2 Experimental Setup

A g4dn.xlarge EC2 instance offered by Amazon Web Services (AWS) was used for all our experiments. Deep learning frameworks such as PyTorch and Hugging Face transformers were used to develop the code-base. The evaluation metric used is macro F-1 score. We experimented with several hyper-parameter configurations by varying the random seed, learning rate, batch size, sequence length and number of epochs. Vanilla cross-entropy loss was used as the loss function in all experiments.

For the course-grained task (hostile/non-hostile), the BERT + Emoji models were trained with a batch-size of 8 and a maximum sequence length of 200. mBERT and Indic-BERT were used as pre-trained language models for this task. The fine-tuning process involved 10 epochs with AdamW as the optimizer and a learning rate of 1e-5. In this case, varying other hyper-parameters had minimal effect on model performance.

On the other hand, two different models were implemented for the fine-grained tasks. Both architectures (Baseline + Emoji and BERT + Bi-LSTM) were fine-tuned for 15 epochs as the augmented datasets take longer to train. Empirically, it was observed that mBERT had superior performance as compared to Indic-BERT for all fine-grained tasks. Furthermore, we used a batch size of 32 and the maximum sequence length was set to 200. Optimization was performed using the AdamW optimizer with a learning rate of 1e-5.

To further improve results for the fine grained tasks, an ensemble of three BERT + Bi-LSTM models was used for each of the tasks. For each fine grained task, models were trained with a random seed number for 15 epochs and a majority vote was taken during inference. For tied results, the output of the first classifier was selected. This gave an overall boost in the performance for the classification tasks for fake and hate. The optimizer and learning rate are the same as the previous task and a batch size of 8 was selected for both training and inference.

6 Results

Table 2 and Table 3 describe the best results we have obtained against the results of the baseline for coarse grained tasks and fine grained tasks respectively. The former was fairly simple to achieve by using BERT with concatenated emoji embeddings. Furthermore, it is worth noting that most hostile posts have emojis in them whereas the non-hostile posts do not. Therefore, using the emoji embeddings gave a better understanding of the sentiment of the tweet and hence improved the F-1 score.

For the fine-grained tasks, adding emoji embeddings ended up reducing the overall performance of our model. For instance, it is hard to use emojis to differentiate between hate and offensive tweets as both may have similar emojis. Stacking a Bi-LSTM layer on top of BERT did increase the performance as LSTM layers can be helpful in capturing additional contextual information. However, for the classification tasks for hate and fake, using an ensemble gave superior results.

Table 2: F-1 scores for the coarse-grained task

Model / Task	Hostile
Baseline claimed	0.9583
Baseline reproduced	0.9552
BERT + Emoji (mBERT)	0.9669
BERT + Emoji (Indic-BERT)	**0.9734**

Table 3: F-1 scores for the fine-grained tasks

Model / Task	Defamation	Fake	Hate	Offensive
Baseline claimed	0.42	0.7741	0.5725	0.6120
Baseline reproduced	0.3869	0.7737	0.5724	0.6153
Baseline + Emoji	0.4414	0.7635	0.5125	0.5739
BERT + Bi-LSTM	**0.4545**	0.7846	0.5337	**0.6178**
Ensemble of BERT + Bi-LSTM	0.4473	**0.8000**	**0.5752**	0.6054

7 Analysis

This section explores possible explanations for our results. Moreover, a detailed error analysis with examples from the test dataset has also been provided.

7.1 Result Analysis

From Section 6, it is evident that the Indic-BERT + Emoji model outperforms the baseline model on the coarse-grained tasks. This implies that the concatenated emoji embeddings help the model in classifying a post as either hostile or non-hostile more efficiently. Since the dataset is balanced in terms of hostile/non-hostile labels (approximately 50-50 split), these results were achieved without any data augmentation techniques.

For the fine-grained tasks, the BERT + Bi-LSTM model and the ensemble classifier clearly outperform the baseline model. In contrast, the Baseline + Emoji architecture does not produce competitive results for most fine-grained tasks. We suspect that the presence of emojis plays a part in confusing the model as several sub-classes may have identical emojis associated with them. As mentioned in the previous section, LSTM layers can improve performance by incorporating additional contextual information. Apart from changes in model architecture, it was observed that back-translating posts having fine-grained labels helps in boosting the F-1 scores. This data augmentation technique proves to be effective in reducing the inherent imbalance present in the dataset as described in Section 3. For the ensemble classifier, the defamation and offensive F-1 scores may have dipped due to the random seeds used. However, adding more models to the ensemble is a potential solution to this.

7.2 Error Analysis

In Figure 4, we can see some examples where our models make classifications errors. Each example represents a different scenario which forces the model to incorrectly classify a post.

In a linguistic sense, the first example seems defamatory and so both models have classified it so. This is not defamatory as it defames traitors and does not point out anyone as a traitor in particular. The second and third posts are classic examples of metaphors and context that can often be confusing for the models. In the second example, a celebrity's (Kangana Ranawat) house is referred as PoK (part of Kashmir under control of Pakistan) and she is being called a terrorist. In very plain translation or interpretation, it would look like a news or a fact. In the fourth example, talking about someone's death and using emojis confused the models to classify it as fake. However, it is interesting to learn the real meaning of the post. The user is wishing for someone's death, hence making it offensive. The last example is a post offending the Indian National Congress Party as a family-run party. Although this is implied through the first line of the post, it does not have offensive language and forces our

Tweet	English Translation	True_label	Prediction BETR+Bi-LSTM	Prediction Ensemble	Task
['हर रोज जान देनी पड़ती हैं, \nसीमा पर पहरेदारों को, \n\nतब भी घर में बैठे-बैठे डर लगता है... \nदेश के गद्दारों को.. 🤫😡']	['Everyday have to die, \ n the guards at the border, \ n \ n even then, sitting in the house is scared ... \ n the traitors of the country..🤫😡']	0	1	1	Defamation
['@KanganaTeam @Sanjay__baroda पीओके में बनी आतंकी के बनकर पर सर्जिकल स्ट्राइक बीएमसी ने कर दी']	['@KanganaTeam @Sanjay__baroda BMC made surgical strike on terrorists made in PoK']	1	0	0	Hate
['उत्तरा नक्षत्र आरहा हैं कूतिया बावरा गई है https://t.co/Q7cF92d0Y3']	['Denebola star is arriving, bitch has gone out https://t.co/Q7cF92d0Y3']	1	0	0	Hate
['गेंडा स्वामी कुत्ते की मौत मर गया\nसाला ये घटना कब तक काल्पनिक रहेगा\n 😢🤔😛🤣😜']	['Unicorn lord died in a very bad way \ n Damn, how long will this incident remain imaginary \ n']	0	1	1	Fake
['तेरे नाम पे शुरू, तेरे नाम पे ख़त्म !!!!\nगांधी परिवार की कांग्रेस , कांग्रेस गांधी परिवार की .....']	['Start on your name, end with your name !!!! \ n Congress of Gandhi family, Congress of Gandhi family .....']	1	0	0	Offensive

Figure 4: Predicted vs Annotated labels for ambiguous sentences

models to make mistakes. To learn such long-tail patterns exhibited by these tweets, our models need much more data than what is available in this dataset.

We analysed many such examples where single model as well as both models make mistakes in classification. We observed that for fine-grained tasks, our models did not learn complex language dependencies due to class imbalance and fewer training samples. We believe that adding URLs and back-translating posts can help in disambiguation of such posts. However, we still need a lot more data to learn these patterns correctly.

8 Future Work

Although our best-performing models outperform the baseline model proposed in [7], surely there is more room for improvement. In this section, we describe potential directions for future work and ideas for improving our results. We observed our model performance stagnate due to fewer training examples and class imbalance. Augmenting the dataset by translating English tweets from the same timeline as well as region (to maintain similar cultural and geographical information) to Hindi could provide better results. Using comments on a tweet as additional data can help with both coarse-grained and fine-grained tasks. For the coarse grained task, if comments are of similar sentiment then it should be non-hostile whereas a hostile post may have differing and strong views. For fine-grained tasks, it is possible that comments can also mention the sentiment. For instance, someone can comment that the post is fake on a fake post.

Furthermore, other ensemble techniques like bagging and boosting may help in improving the F-1 scores for the fine-grained tasks. Finally, we believe that our trained models can be used to design an end-to-end pipeline for hostile language detection on social media.

Acknowledgments

We would like to acknowledge the guidance of Ojasv Kamal and Adarsh Kumar, authors of [7], in helping us understand the baseline model. We also thank Professor Graham Neubig, Pengfei Liu and Ritam Dutt for providing valuable feedback and suggestions.

References

[1] Ibrahim Abu Farha and Walid Magdy. Multitask learning for Arabic offensive language and hate-speech detection. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection*, pages 86–90, Marseille, France, May 2020. European Language Resource Association.

[2] Sai Saketh Aluru, Binny Mathew, Punyajoy Saha, and Animesh Mukherjee. Deep learning models for multilingual hate speech detection, 2020.

[3] Pinkesh Badjatiya, Shashank Gupta, Manish Gupta, and Vasudeva Varma. Deep learning for hate speech detection in tweets. *Proceedings of the 26th International Conference on World Wide Web Companion - WWW '17 Companion*, 2017.

[4] Mohit Bhardwaj, Md Shad Akhtar, Asif Ekbal, Amitava Das, and Tanmoy Chakraborty. Hostility detection dataset in hindi, 2020.

[5] Varad Bhatnagar, Prince Kumar, Sairam Moghili, and Pushpak Bhattacharyya. Divide and conquer: An ensemble approach for hostile post detection in hindi, 2021.

[6] Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. Automated hate speech detection and the problem of offensive language. In *Proceedings of the 11th International AAAI Conference on Web and Social Media*, ICWSM '17, pages 512–515, 2017.

[7] Ojasv Kamal, Adarsh Kumar, and Tejas Vaidhya. Hostility detection in hindi leveraging pre-trained language models, 2021.

[8] Animesh Koratana and Kevin Hu. Toxic speech detection. In *Neural Inf. Process. Syst.*, page 9, 2018.

[9] Ioannis Mollas, Zoe Chrysopoulou, Stamatis Karlos, and Grigorios Tsoumakas. Ethos: an online hate speech detection dataset, 2020.

[10] Jihyung Moon, Won Ik Cho, and Junbum Lee. Beep! korean corpus of online news comments for toxic speech detection, 2020.

[11] P. K. Roy, A. K. Tripathy, T. K. Das, and X. Z. Gao. A framework for hate speech detection using deep convolutional neural network. *IEEE Access*, 8:204951–204962, 2020.

[12] Sayar Ghosh Roy, Ujwal Narayan, Tathagata Raha, Zubair Abid, and Vasudeva Varma. Leveraging multilingual transformers for hate speech detection, 2021.

[13] Chander Shekhar, Bhavya Bagla, Kaushal Kumar Maurya, and Maunendra Sankar Desarkar. Walk in wild: An ensemble approach for hostility detection in hindi posts, 2021.

[14] Ziqi Zhang and Lei Luo. Hate speech detection: A solved problem? the challenging case of long tail on twitter, 2018.