Hostile Post Detection in Hindi: IITK@CONSTRAINT 2021

CS685A End-Term Project Report - Group 25

Prof. Arnab Bhattacharya

Priya Mishra (170508)
priyamis@iitk.ac.in
Sagnik Mukherjee(170606)
sagnikm@iitk.ac.in
Raunak Shah (170560)
raunaks@iitk.ac.in
Yash Maheshwari (170817)
yashma@iitk.ac.in

Abstract

We have witnessed a sudden increase in the accessibility of the internet in India in the recent decade. This has resulted in various social media platforms being accessible to users in native languages which has further ensued the rise of online hate speech in Hindi. In this project, we aim to tackle this problem of online hate speech in Hindi and have participated in the Hostile Post Detection in Hindi shared subtask in CONSTRAINT 2021. We work on a dataset consisting Hindi posts from Facebook and Twitter and classify them into the categories Non-Hostile, Fake, Hate, Offensive and Defamation. We employ various text pre-processing techniques and evaluate the performance of models like RNN and Bert on the processed data. We discuss our methodology and observations in detail in this report.

1 Introduction & Motivation

With the continuous rise in the number of people using Internet and social media, use of hate speech online has become a great problem for online moderators as well as internet users. This problem needs to be tackled to create a safe space for users online while at the same time extreme measures cannot be taken which would result in total censorship of social media platforms. The differences between hostile and non-hostile speech may sometimes be very minute which makes the task of differentiating difficult for human social media moderators. This makes the problem more difficult as we have to take additional care to not mark any non-hostile post as hostile. At the same time, the use of hate speech can lead to violence against minority groups[1], cyberbullying and ultimately makes the internet an unsafe place for its users. During major events like the 2021 US Presidential elections or the Covid-19 crisis[2], the use of online hate speech is expected to increase manifold[3] which leads to young users being exposed to explicit content and its normalisation on social media. This creates another problem for users.

As more and more people gain access to internet and therefore social media platforms in India, the posts in native languages like Hindi, Bengali, Marathi have also increased substantially[4]. Posts written in regional languages often evade online hate speech filters which are designed to work on popular resource rich languages like English. This further complicates the challenge for languages like Hindi for which the amount of datasets to work on is very low and not much research has been performed.

To combat this problem, a shared subtask has been released on the AAAI 2021 workshop (Constraint) for hostile post detection in Hindi language. In our project, we will focus on classifying Facebook, Twitter or Whatsapp posts written in Hindi into hate speech, fake news, offensive, defamation, and non-hostile through models like RNN, BERT and other statistical classifiers.

2 Related Work

This issue of use of hate speech and offensive language has been troubling internet users since the beginning of the internet. The earliest work in the field of abuse detection on internet involved the use of C4.5 decision tree generator to get feature vectors to detect abusive messages (Spertus,

[1] https://phys.org/news/2019-10-online-speech-crimes-minorities.html

[2] https://phys.org/news/2020-11-young-people-exposed-online-covid.html

[3] https://l1ght.com/Toxicity_during_coronavirus_Report-L1ght.pdf

[4] 'Non-English tweets are now 50% of the total': Twitter India MD - http://www.ecoti.in/bR4s3Y33

1997). Since then, the field has advanced significantly with the introduction of new machine learning techniques and n-gram modelling. (Yin et al., 2009) were the first to use supervised learning methods for hostile post detection on social media on the Web 2.0. They used SVMs along with sentiment-based features to classify the posts. Various similar works involving neural networks have also been done in this field in English.

This detection problem becomes difficult when English is code-mixed with regional languages like Hindi to give rise to new writing styles like *Hinglish* since words, phrases and sentences from different languages may co-exist within a post, and the models are required to recognize and process these simultaneously. This new style is being used extensively on social media in India. (Bohra et al., 2018) have worked on this new style by creating a dataset from scratch. they used SVMs on character n-grams, word n-grams, punctuation, lexicon and negations features. Further, (Mathur et al., 2018) have also worked on this style using CNNs.

However, these problems have now become easier to solve with the development of novel machine learning models like Universal Language Model Fine-Tuning (ULMFiT) (Howard and Ruder, 2018), Embedding from Language Models (ELMO) (Peters et al., 2018), OpenAI's Generative Pre-trained Transformer (GPT) (Radford, 2018) and Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2018). Out of these, BERT has proved to significantly outperform the other models. Therefore, we'll be using a fine-tuned BERT model here.

With the introduction of *Devanagari* script based keyboards and writing softwares, the spread of hostile posts written in pure Hindi has also grown significantly since they have higher potential to attract a local readers' attention. The work here is in a nascent stage and recently received a boost through the Shared Task on Hate Speech and Offensive Content Identification in Indo-European Languages in HASOC 2019 [5]. (Mishra and Mishra, 2019) is a notable work in this workshop. They used monolingual and multilingual BERT based neural networks and achieved the best results for Hindi subtask. Further, (Rani et al., 2020) have also worked on the code-mixed and pure Hindi datasets mixed together using KNN, SVM, MNB and Decision trees with Term frequency (TF) weighting as a feature. Although, they achieved the best F1 score through character based CNN described by (Zhang et al., 2015). We have tried to build up on these works and have focused on the BERT models which are best in class for NLP tasks.

[5]Hate Speech and Offensive Content Identification in Indo-European Languages (HASOC2019): https://hasocfire.github.io/hasoc/2019/index.html

3 Dataset Statistics

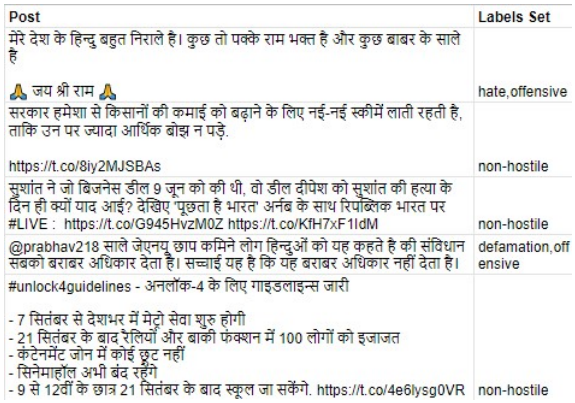
The initial dataset provided contains the training and the validation sets. The datasets contain posts in Hindi Devanagri Script collected from Twitter and Facebook. Each post is given a Unique ID and a label set. The set of valid labels are fake news, hate speech, offensive, defamation, and non-hostile posts. This is a multi-label, multi-class classification. The hostile posts can belong to one or more of these hostile classes. However, the non-hostile posts cannot have any other label. The distribution of various classes in training and validation sets is shown in Table 1.

Table 1: Distribution of Training and Validation sets

Label	Train	Validation
Non-Hostile	3050	435
Fake	1144	160
Hate	792	110
Offensive	742	103
Defamation	564	77
Total	5728	811

Definitions of the class labels as per the shared task:

- **Fake News:** A claim or information that is verified to be not true.
- **Hate Speech:** A post targeting a specific group of people based on their ethnicity, religious beliefs, geographical belonging, race, etc., with malicious intentions of spreading hate or encouraging violence.
- **Offensive:** A post containing profanity, impolite, rude, or vulgar language to insult a targeted individual or group.
- **Defamation:** A mis-information regarding an individual or group.
- **Non-hostile:** A post without any hostility.



Post	Labels Set
मेरे देश के हिन्दु बहुत निराले है। कुछ तो पक्के राम भक्त है और कुछ बाबर के साले है 🙏 जय श्री राम 🙏	hate,offensive
सरकार हमेशा से किसानों की कमाई को बढ़ाने के लिए नई-नई स्कीमें लाती रहती है, ताकि उन पर ज्यादा आर्थिक बोझ न पड़े. https://t.co/8iy2MJSBAs	non-hostile
सुशांत ने जो बिजनेस डील 9 जून को की थी, वो डील दीपेश को सुशांत की हत्या के दिन ही क्यों याद आई? देखिए 'पूछता है भारत' अर्नब के साथ रिपब्लिक भारत पर #LIVE : https://t.co/G945HvzM0Z https://t.co/KfH7xF1IdM	non-hostile
@prabhav218 साले जेएनयू छाप कमिने लोग हिन्दुओं को यह कहते है की संविधान सबको बराबर अधिकार देता है। सच्चाई यह है कि यह बराबर अधिकार नहीं देता है।	defamation,offensive
#unlock4guidelines - अनलॉक-4 के लिए गाइडलाइन्स जारी - 7 सितंबर से देशभर में मेट्रो सेवा शुरु होगी - 21 सितंबर के बाद रैलियों और बाकी फंक्शन में 100 लोगों को इजाजत - कंटेनमेंट जोन में कोई छूट नहीं - सिनेमाहॉल अभी बंद रहेंगे - 9 से 12वीं के छात्र 21 सितंबर के बाद स्कूल जा सकेंगे. https://t.co/4e6lysg0VR	non-hostile

Figure 1: First five original data samples from the training dataset

4 Data Preprocessing

Since our dataset contains tweets and facebook posts, the initial pre-processing includes the following steps:

- Removing all URLs
- Removing usernames
- Removing emoticons
- Removing some punctuation marks, extra white spaces and any other unnecessary characters

We decided to keep hashtags in this pre-processed text as in many posts, the hashtags may contain some useful information which helps us in our prediction task. We also preserved exclamation marks for the same purpose.This initial data pre-processing was done using simple string functions in Python.

Figure 2: First five data samples from the training dataset after the initial pre-processing

Then, we tokenize the cleaned sentences. For this we have used the Indic NLP library (Kunchukuttan, 2020), which provides a tokenizer for multiple Indian languages including Hindi.

After we get the tokenized array, we proceed to remove the stop words from this array. Removing stop words may not have a major effect on the model's performance (Ghag, 2015) but helps us in reducing the size of the dataset that we are working with which helps in reducing required storage space as well as computation time.

Once we have the cleaned text array, we proceed to lemmatise the words. We have used the Stanza (Qi et al., 2020), a python NLP package by StanfordNLP (Manning et al., 2014) to complete this task. Lemmatisation helps us in shortening the words while preserving their inherent meaning in the sentence so that we can extract the same representation form different words which have the same meaning.

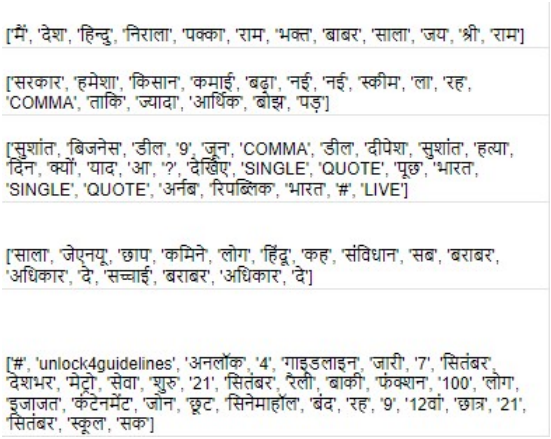


Figure 3: First five data samples from the training dataset after initial pre-processing, tokenisation, removing stop-words and lemmatisation

We also tried Data Augmentation to deal with the class imbalance problem in our dataset. Here, we take the final array after the lemmatisation step and find synonymns for all 'common noun' and 'adjective' words on the array and for each sample in the dataset, construct another sample by replacing the common nouns and adjectives with their synonyms. For this task, we have used a Python based API to access Indian language WordNets (PyIWN) (Panjwani et al., 2018). We discuss the results obtained from Data Augmentation later in the report.

5 Methodology

Broadly the task can be described as a multi-label classification task. The challenge is posed by

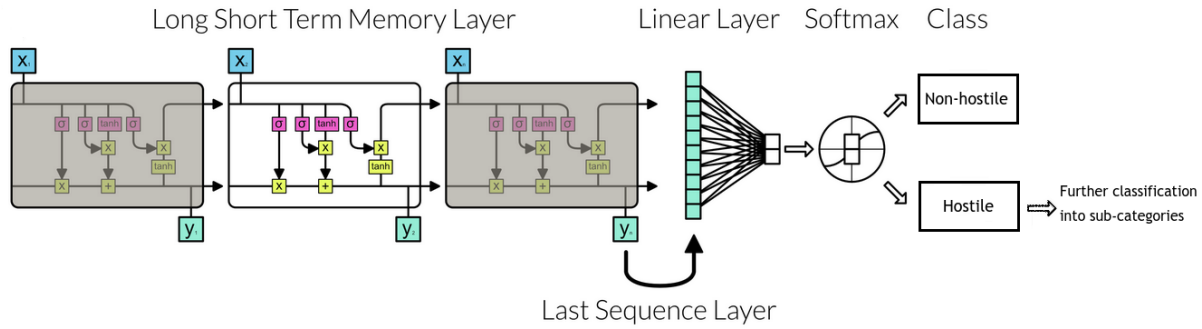


Figure 4: Example of the baseline: Classification into hostile/non-hostile categories using LSTM + Dense layer. Further sub-classification into fake, defamation, hate, and offensive can be done for the hostile posts using the generated sequences from the LSTM

the size of the dataset and the language of interest. Hindi being a morphologically rich language, text mining algorithms are often not able to produce good results. Also, the amount of variability makes it almost impossible for a simple model to capture the grammar of the language. We therefore have used pretrained large language models for that purpose. The overall approach is to use a neural network, followed by a classifier that can be used to classify the sentence into multiple classes. With this idea, the architectures we have used are as follows.

5.1 Baseline: RNN + NN Classifier

Since the organisers had not provided us with any base line models at the start of the competition, we chose a simple RNN based model as our baseline. We used 300 dimensional glove embeddings and built a linear layer on top of our Bi-LSTM. Even this naive model reached a weighted validation score of 0.66 on the validation data. Architectural details of the model are described in Fig 4 and Section 6.1.

However, the competition baselines are extremely strong than what we estimated it to be. The organizers have used the multilingual bert model as the baseline. They keep the output of the last layer as the embedding vectors for each word in a sentence, and their average as a sentence representation. Further they built Random Forest, SVM and MLP based classifiers on top of those. Further details can be explored at the hindi dataset paper (Bhardwaj et al., 2020).

5.2 BERT fine-tuning

We deployed several techniques for finetuning bert for our task. However, we noticed that different finetuning schemes for the same bert variant, didn't affect the scores by a significant margin.

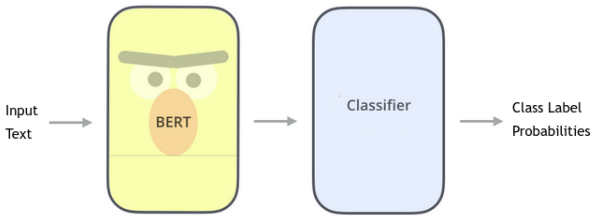


Figure 5: Finetuning using BERT. Image source: (Alammar, 2018)

- **MLP:** We built a standard 1 or 2 layer MLP on top of the BERT model. Across experiments we either classified the [CLS] token or the average for the entire sentence. For more than 2 layers of MLP, the results get worse (Due to overparametrization). The dimension of the intermediate layer of the MLP was a hyperparameter, that was tuned manually.

- **RNN Variant:** In this setup, we built an additional RNN based model, LSTM or GRU, on top of the BERT model. And after that we had a fully connected dense layer as the final classifier of the model. On most of the experiments, GRU performed better than LSTMs.

- **CNN:** In this technique, output from all layers of bert are used instead of just the final one. We had a CNN kernel for each output layer of bert, and did a maxpooling operation on them. The 12 maxpooled outputs are then concatenated and passed through a linear layer to classify.

5.2.1 Data Augmentation

In the confusion matrices obtained from our experiments for various BERT based models, we observed that the results were not good for the classes defamation, hate and offensive. This problem can be attributed to the sparsity problem in the training dataset for these classes. Further, there are very fine distinctions in these classes and classifying posts into these classes is a difficult task for humans too due to the fine distinction. To solve the problem of class imbalance, we tried Data Augmentation where we created new data points by slightly modifying the already present data points. We tried two variants of Data Augmentation; first we tried creating a new data point for each data point irrespective of their labels in the training dataset which essentially doubled the size of the dataset. In the second variant, we scaled up the data points based on the distribution of their labels as given in the Table 1. Here, we created 1, 2, 2 and 3 new data points for each data point with the labels fake, hate, offensive and defamation. We experimented using this data with the Hindi BERT + Linear model. We discussed the creation of a new data point in the section 4 and discuss the results obtained in the section 6.2.1.

5.2.2 Output Threshold Optimisation

After conducting the experiments with various variants of BERT, we performed output threshold optimisation for the end layer in our model where we threshold the output probabilities of the model for each class. First, we tried uniform thresholding for the Hindi Bert model where we applied the same threshold to probability outputs for all the 5 classes. We performed grid search for the same in the range $[0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9]$ and constructed the Precision-Recall curve and the F1-Threshold curve to find the optimum uniform threshold. The same are illustrated in Figure 6

Even after this optimisation, we observed from the confusion matrices that the problem of class imbalance was still affecting the results. So, we decided to proceed with setting different thresholds for different classes in the output. We performed grid search for the same and found the optimum thresholds: 0.2, 0.3, 0.3, 0.3, 0.1 for the Non-hostile, hate, offensive, defamation and fake classes respectively. The results for the same are highlighted in the section 6.2.2 and the confusion matrices are illustrated in Figure 11.

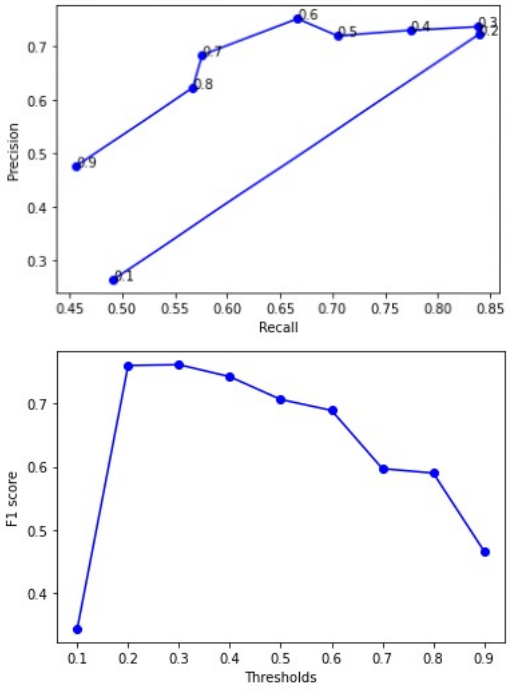


Figure 6: Precision-Recall curve and F1-Threshold curve for uniform thresholding on Hindi BERT+linear model

6 Results & Discussion

6.1 Baseline Results

For the baseline RNN architecture we used LSTM and GRU models. The detailed results are shown in Table 2. We experimented with varying layers, and varying thresholds for different classes. For both the models, we used one, two or four layers. We observe that for both LSTM and GRU models, we obtain the best performance using two layers. The performance drops when we increase this to four layers. We used 128 hidden units for all models. The best performance (F1 score) using a default threshold was 0.5856 and 0.5904 for LSTM and GRU models, respectively.

Next, we used precision-recall curves to identify the optimum threshold for each label. Using different thresholds for each class improved the performance of both the models. The validation F1 score increased to 0.6655 and 0.6231 for LSTM and GRU models respectively.

Among all the experiments for the baseline, the best performance was obtained using BiLSTM with 2 layers and varying thresholds. The vaildation precision, recall and F1 score for this model are 0.6976, 0.6486 and 0.6655 respectively.

Model	RNN Layers	Weighted Metrics (Train)			Weighted Metrics (Validation)		
		Precision	Recall	F1 Score	Precision	Recall	F1 Score
BiLSTM + Linear	1	0.8402	0.6081	0.6671	0.6154	0.478	0.5248
BiLSTM + Linear	4	0.5208	0.4741	0.4954	0.4933	0.4395	0.4633
BiLSTM + Linear	2	0.7657	0.6387	0.6621	0.6751	0.5514	0.5856
BiLSTM + Linear (Varying Thresholds)	2	0.7691	0.7462	0.7505	**0.6976**	**0.6486**	**0.6655**
BiGRU + Linear	1	0.7663	0.6235	0.6671	0.6511	0.5412	0.5707
BiGRU + Linear	4	0.7959	0.5931	0.6644	0.6947	0.5006	0.5717
BiGRU + Linear	2	0.7809	0.6454	0.6838	0.6874	0.5537	0.5904
BiGRU + Linear (Varying Thresholds)	2	0.8124	0.6934	0.5709	**0.7199**	**0.5876**	**0.6231**

Table 2: Fine-grained results for RNN-baseline models

Language Model	Classifier	Weighted Metrics (Train)			Weighted Metrics (Validation)		
		Precision	Recall	F1 Score	Precision	Recall	F1 Score
Hindi BERT	Linear	0.8039	0.8098	0.8018	0.7182	0.6994	0.7043
	GRU	0.8833	0.8159	0.8201	0.7738	0.6915	0.7051
	CNN	0.8738	0.8536	0.8597	0.7269	0.7129	0.7178
Hindi TPU Electra	Linear	0.7456	0.7299	0.7376	0.6768	0.6441	0.6598
	LSTM	0.7594	0.7603	0.7374	0.7196	0.6859	0.6915
	CNN	0.8740	0.7703	0.7898	0.7277	0.7209	0.7200
TRAC A ALL Multilingual BERT	Linear	0.8946	0.7058	0.7506	0.7986	0.6406	0.6841
	GRU	0.8580	0.7735	0.7924	0.7700	0.6926	0.7136
	CNN	0.8538	0.7428	0.7368	0.7794	0.7266	0.7368
TRAC A Hindi BERT Cased	Linear	0.8706	0.7342	0.7670	0.8187	0.6779	0.7124
	GRU	0.8343	0.7698	0.7845	0.7480	0.696	0.7101
	CNN	0.8829	0.7994	0.8340	0.7726	0.7119	0.7396
XLM Roberta	Linear	0.8843	0.8687	0.8674	0.7887	0.7630	0.7710
	CNN	0.9493	0.8978	0.9183	0.8006	0.7559	0.7759

Table 3: Fine-grained results for BERT-based models

6.2 BERT-based Models

We experimented across various BERT based models and multiple classifiers on top of them as well. Our results are summarized on table 3. The XLM model showed the best results on the validation data with only 5 epochs of training, and the CNN based classifier was the best performer amongst different classifiers for all BERT based models.

6.2.1 Data Augmentation

We observed a degradation in the F1 scores after performing Data Augmentation. F1 scores observed for the first and second variant were 0.8851 for Train, 0.6632 for Validation and 0.7268 for Train, 0.6077 for Validation respectively. We are able to observe the model over-fitting on the Augmented dataset from the large differences between the F1 scores on the training and validation datasets and attribute the poor results to the same. The confusion matrices for this experiment are shown in Figure 11. Here we can observe that the performance for this technique is worse than that obtained on the original test dataset for some classes and comparable in others. Therefore, we don't observe much improvement here.

6.2.2 Output Threshold Optimisation

We observed an improvement in the F1 scores after the threshold optimisation. The F1, precision and recall scores were **0.8455**, 0.7984, 0.9615 for the training dataset and **0.7685**, 0.7298, 0.8757 for

the validation dataset. Significant improvement can be seen after this optimisation in the performance of the Hindi BERT + linear model, comparing the same from the table 3. The confusion matrices for the same are plotted in Figure 11. Through the same, we can observe that this method is better than the other two methods on the Hindi BERT+Linear model for all classes except the Hate class.

6.2.3 Convergence Curves

In the following figures, accuracy refers to F1 score.

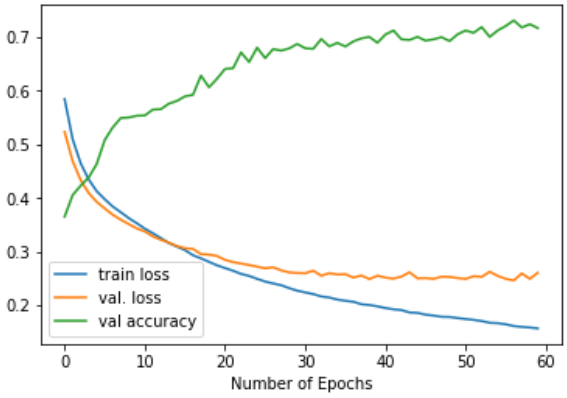


Figure 7: Convergence curves for Hindi BERT model with Linear Classifier

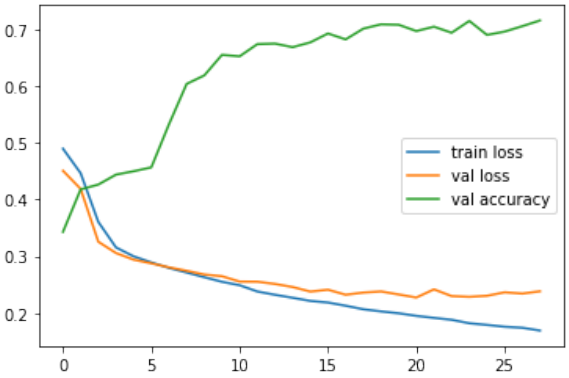


Figure 8: Convergence curves for Hindi Electra model with CNN Classifier

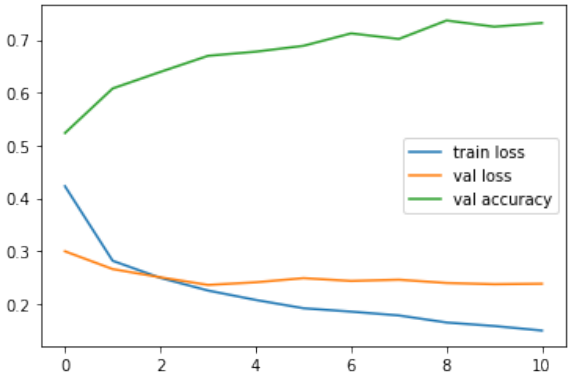


Figure 9: Convergence curves for TRAC A ALL Multilingual model with CNN Classifier

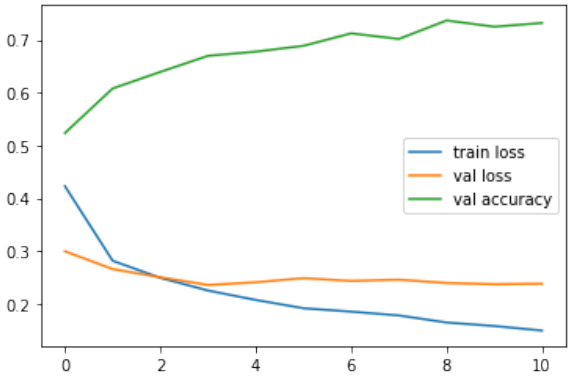
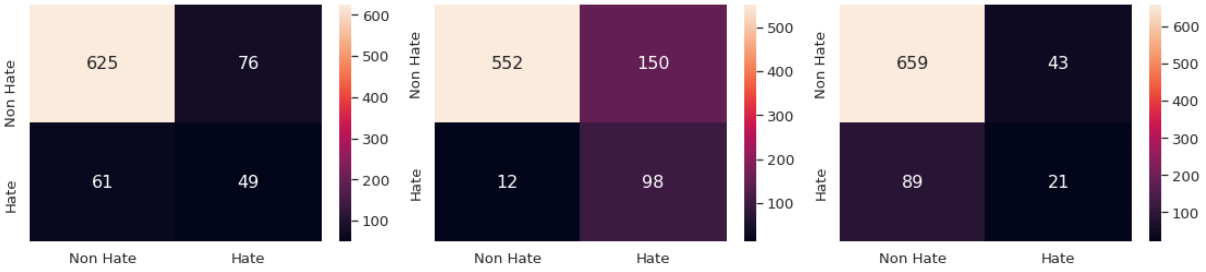


Figure 10: Convergence curves for TRAC A Hindi model with CNN Classifier

7 Future Directions

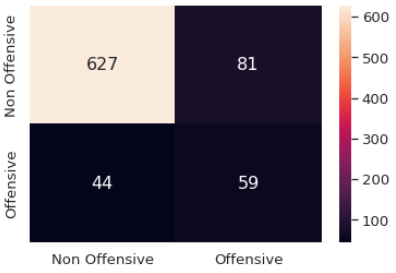
The main objective of this task is to fight the ”COVID 19 Infodemic” and that can’t be limited to building a model. The immediate step after this work should be to build a web-based service to classify the tweets. As we have already seen - The success rate of the classes ”hostility” and ”fake” is extremely high. Which means spread of fake news can be easily prevented with this architecture. Also, a coarse grained evaluation will be highly successful. However, for further fine-grained evaluation might not be that successful. The classes ”defamation”, ”hate” and ”offensive” don’t have a well-defined boundary even for humans. Also, the dataset had an issue of class imbalance, which made it really hard to develop decent neural network based models, since all such models are heavily dependant on the amount of data. From that perspective, it is necessary to build datasets with better balance between different classes.



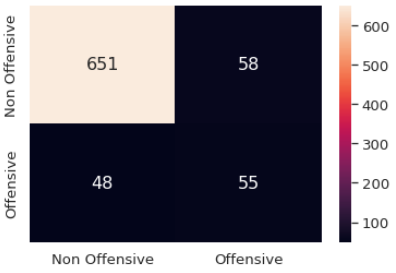
(a) Hate Confusion Matrix

(b) Hate Confusion Matrix

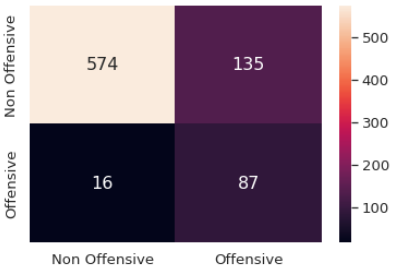
(c) Hate Confusion Matrix



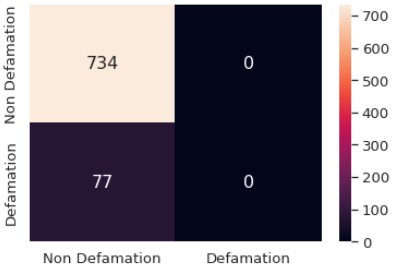
(d) Offensive Confusion Matrix



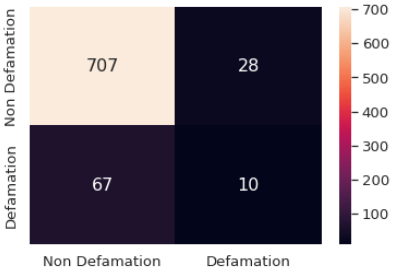
(e) Offensive Confusion Matrix



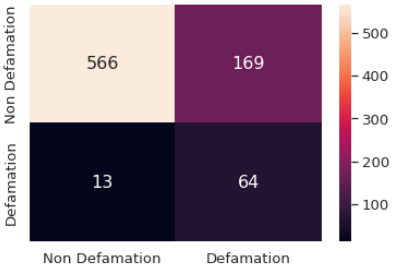
(f) Offensive Confusion Matrix



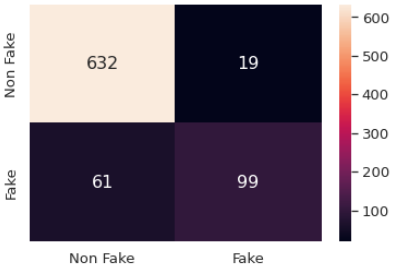
(g) Defamation Confusion Matrix



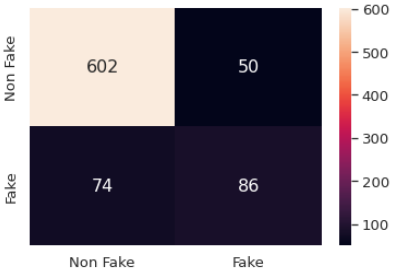
(h) Defamation Confusion Matrix



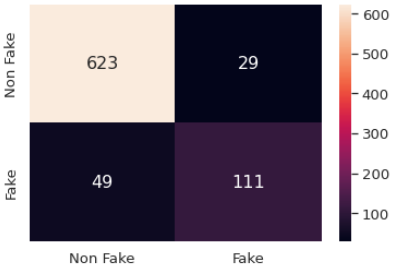
(i) Defamation Confusion Matrix



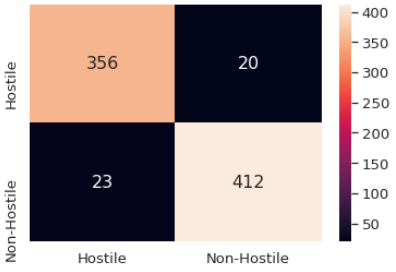
(j) Fake Confusion Matrix



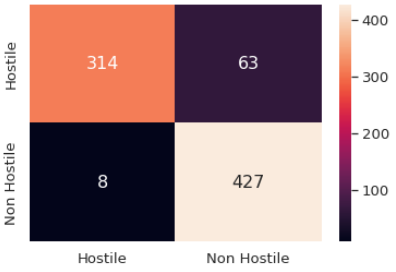
(k) Fake Confusion Matrix



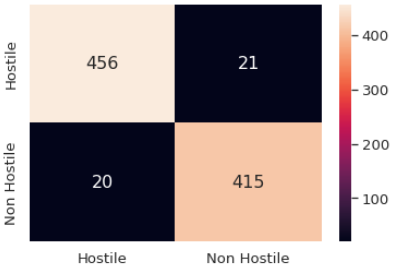
(l) Fake Confusion Matrix



(m) Hostility Confusion Matrix



(n) Hostility Confusion Matrix



(o) Hostility Confusion Matrix

Figure 11: Confusion Matrices on Validation Set for the model Hindi BERT + Linear. The first column is for the original dataset, the middle column is for training on the augmented dataset, and the last column is after output threshold optimisation on the original dataset. The contrast between the 3 techniques can be seen here in the confusion matrices.

Note: Top left is True Negative, Top right is False Positive, Bottom left is False negative and Bottom right is True Positive in these confusion matrices.

References

[Alammar2018] Jay Alammar. 2018. The illustrated bert, elmo, and co. December.

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

[Bohra et al.2018] Aditya Bohra, Deepanshu Vijay, Vinay Singh, Syed Sarfaraz Akhtar, and Manish Shrivastava. 2018. A dataset of Hindi-English code-mixed social media text for hate speech detection. In *Proceedings of the Second Workshop on Computational Modeling of People's Opinions, Personality, and Emotions in Social Media*, pages 36–41, New Orleans, Louisiana, USA, June. Association for Computational Linguistics.

[Devlin et al.2018] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

[Ghag2015] Kranti Ghag. 2015. Comparative analysis of effect of stopwords removal on sentiment classification. pages 1–6, 09.

[Howard and Ruder2018] Jeremy Howard and Sebastian Ruder. 2018. Fine-tuned language models for text classification. *CoRR*, abs/1801.06146.

[Kunchukuttan2020] Anoop Kunchukuttan. 2020. The IndicNLP Library. `https://github.com/anoopkunchukuttan/indic_nlp_library/blob/master/docs/indicnlp.pdf`.

[Manning et al.2014] Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, pages 55–60.

[Mathur et al.2018] Puneet Mathur, Rajiv Shah, Ramit Sawhney, and Debanjan Mahata. 2018. Detecting offensive tweets in Hindi-English code-switched language. In *Proceedings of the Sixth International Workshop on Natural Language Processing for Social Media*, pages 18–26, Melbourne, Australia, July. Association for Computational Linguistics.

[Mishra and Mishra2019] Shubhanshu Mishra and Sudhanshu Mishra. 2019. 3Idiots at HASOC 2019: Fine-tuning Transformer Neural Networks for Hate Speech Identification in Indo-European Languages. In *Proceedings of the 11th annual meeting of the Forum for Information Retrieval Evaluation*, FIRE 2019 Working Notes, pages 208–213, Kolkata, India, December. CUER Workshop Proceedings.

[Panjwani et al.2018] Ritesh Panjwani, Diptesh Kanojia, and Pushpak Bhattacharyya. 2018. pyiwn: A python-based api to access indian language wordnets. In *Proceedings of the Global WordNet Conference*, volume 2018.

[Peters et al.2018] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *CoRR*, abs/1802.05365.

[Qi et al.2020] Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A Python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*.

[Radford2018] A. Radford. 2018. Improving language understanding by generative pre-training.

[Rani et al.2020] Priya Rani, Shardul Suryawanshi, Koustava Goswami, Bharathi Raja Chakravarthi, Theodorus Fransen, and John Philip McCrae. 2020. A comparative study of different state-of-the-art hate speech detection methods in Hindi-English code-mixed data. In *Proceedings of the Second Workshop on Trolling, Aggression and Cyberbullying*, pages 42–48, Marseille, France, May. European Language Resources Association (ELRA).

[Spertus1997] E. Spertus. 1997. Smokey: Automatic recognition of hostile messages. In *AAAI/IAAI*.

[Yin et al.2009] Dawei Yin, Zhenzhen Xue, Liangjie Hong, Brian D Davison, April Kontostathis, and Lynne Edwards. 2009. Detection of harassment on web 2.0. *Proceedings of the Content Analysis in the WEB*, 2:1–7.

[Zhang et al.2015] Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *CoRR*, abs/1509.01626.