 doi:10.20944/preprints202011.0646.v1

Article

Online Multilingual Hate Speech Detection: Experimenting with Hindi and English Social Media

Neeraj Vashistha [1], **Arkaitz Zubiaga** [2]

[1] Queen Mary University of London; ec19471@qmul.ac.uk
[2] Queen Mary University of London; a.zubiaga@qmul.ac.uk

Abstract: The exponential increase in the use of the Internet and social media over the last two decades has changed human interaction. This has led to many positive outcomes, but at the same time it has brought risks and harms. While the volume of harmful content online, such as hate speech, is not manageable by humans, interest in the academic community to investigate automated means for hate speech detection has increased. In this study, we analyse six publicly available datasets by combining them into a single homogeneous dataset and classify them into three classes, abusive, hateful or neither. We create a baseline model and we improve model performance scores using various optimisation techniques. After attaining a competitive performance score, we create a tool which identifies and scores a page with effective metric in near-real time and uses the same as feedback to re-train our model. We prove the competitive performance of our multilingual model on two langauges, English and Hindi, leading to comparable or superior performance to most monolingual models.

Keywords: social media; hate speech; text classification

1. Introduction

Hate Speech is a characterisation of communication which is 'hateful', controversial and generates intolerance and in some forms is divisive and demeaning. There is no legal definition of hate speech, but on several accounts accepted meaning of the term deals with communication in speech, behaviour or writing, remarks which are pejorative or discriminatory with reference to a person or a group of persons, either directly or indirectly. Such remarks are based on their religion, ethnicity, nationality, descent, race, colour, gender or other identity factor [1].

Many countries have adopted frameworks and new laws have been constituted but due to the pervasive nature of online communications, only 3% of malicious communication offenders are charged [2]. This is because of lack of clarity and certainty in this area. Several of these frameworks prohibit the incitement to discrimination, hostility and violence rather than prohibiting hate speech. There is a need for quick identification of such remarks and an automatic system which can identify and take measures to prevent the instigation and incitement.

There are several examples of hate speech such as

Reminding feminists world-wide that they are only mad because they're ugly...

Good morning WHITE people! F you N**

which either implicitly or explicitly target an individual or a group of individuals and inflict mental pain that may eventually cause of social revolts or protests.

In this work, we study the existing methods designed to tackle hate speech. We have curated the data from the most prominent social media platform, Twitter. We also combine the existing datasets into one, as these are pre-annotated and previous researchers have used these to target several cases of hate speeches. Individually, these are small datasets but after we combine the existing datasets into one dataset, we get a large corpus of hate speech sequences. This dataset is a combination of the hateful content of almost 5 different categories, such as sexual orientation, religion, nationality, gender,

 doi:10.20944/preprints202011.0646.v1

and ethnicity. We create models which classify the content of text as hateful, abusive, or neither. The total size of the dataset is around seventy six thousand samples and due to the high variance and low bias we are avoiding sub-categorisation of hate classes.

We also study some state-of-the-art models which claim to give superior accuracy. Since these models are built on specific languages (English or Hindi), we utilise the work and propose our model which is multilingual. Finally, we use our optimised models to create a simple tool which identifies and scores a page if hateful content is found and uses the same as the feedback to re-train the model. While the vast majority of previous works have investigated the development of hate speech detection models for specific languages [3], here we propose a multilingual model which we experiment in two languages, English and Hindi, leading to competitive performance and superior to most monolingual models.

The main contribution of this work are as follows:

- Creating a system which is trained on a sufficiently large corpus of hate speech text.
- A system which is multilingual, can work on different languages or another language be added and trained quickly using our transfer learning models .
- A system which can learn new form of hateful remark or zipfian nature of language by re-training in an online environment.
- The resulting system has models which are simple, lightweight and are optimised to be used in an near-real-time online environment.

In this way the system holds a potential as a product for safer social media utilization as well as reduces the need for human annotators and moderators to tag disturbing online messages.

The rest of the paper is organised as follows: an overview of hate speech datasets and existing models is provided in the Related Work section. In the Model section, we critically analyse the existing models and discuss their limitations. We also discuss our model and several optimisations we performed to achieve a desirable performance score. Furthermore, we talk about the feedback mechanism for the optimised model and its usage in an online environment.

2. Related Work

Here we discuss related work on hate speech datasets, hate speech detection models and the different approaches. This serves as a motivation for our work and helps us to bridge our study with existing research available.

2.1. Related Datasets

As mentioned earlier, the six publicly available datasets [4–9] are manually curated and are of modest size. In the below text, we discuss the characteristics and the generation process of each of these datasets. Using these annotated datasets we create our own resource of hate speech sequences.

2.1.1. Data Gathering process

The data in online diaspora can be curated from various technology companies and social media platform providers such as Facebook, Google, Internet Service Provider Association, Oath, Twitter, Snap Group Limited etc. Twitter's Public API, with its easy to use, highly availability and accessibility, it is one of the most sought and targeted online social media platforms. We can set up a mechanism and relevant data for hate speech can be sourced. The dataset we are using in this study comes from Twitter and is labelled by the annotators into pre-specified categories and subcategories of hate speech.

The examples sourced from Twitter are based on keyword matching with terms which have strong connection with hate. These are annotated and tagged as abusive or hateful. This task has to be done manually and is very time consuming. The same process has been adopted for code-mixed and pure Hindi language. Since the sample space of [9] and [4] is small, we have tried to update these datasets with our own curated samples.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 25 November 2020 doi:10.20944/preprints202011.0646.v1

To describe the multilingual aspect, the dataset consists of three different types of target language, English, Hindi and Code Mix Hindi. Below 1 describe various attributes of the dataset. Although there

Table 1. Multilingual Dataset Attributes

Language	Attributes		
	Total Records	*Vocab Size*	*Max Seq. Len*
English-EN	61,319	23,216	89
Hindi-HI	9,330	6,082	104
Hindi Code-Mix	3,161	6,094	48

are some limitations such as small sample text size of 160 characters or under representation of all forms of hate speech sequences but this does not, in any way, limit the use of data. Other issues related to this kind of dataset is the filtering of unwanted details like email address, URL, date, phone number, username etc due to which textual and contextual information is lost. We also try that the collation of different datasets doesn't impact the attributes. As these datasets are tagged by different authors as per their own perception and with the daily evolution of slang and jargon, evaluation becomes a tedious task.

2.1.2. Existing Datasets

HASOC2019: The dataset presented in [4] shared task on hate speech and offensive content identification in Indo-European language contain around 7005 post in English and 9330 post in Hindi language, with 36% and 52% abusive content in respective languages. The task is focused on three sub-tasks,

- if the tweet text is either hate or offensive or neither
- if tweet text is either hate, offensive or profane
- if the tweet text is targeted towards an individual/group or is untargeted

For this study we are only considering the data presented for the first sub-task.

TDavidson et al: The dataset presented in [5] research includes tweets labeled by crowdflower workers into three categories: hate speech, offensive, or neither. There are 24,802 English twitter text instances with only 6% abusive content.

ElSherif et al: The dataset presented in [6] research is procured from twitter and it is categorised into 7 different categories of Hate. The data is being sought from [10] and [5] with their own annotation from No Hate Speech Movement ([1]) and Hatebase [2]. The total number of samples is 10760 with total abusive content around 10%. This dataset is in English language.

Ousidhoum et al: Here the author has categorised twitter data into 46 different sentiments of hateful, offensive, fearful, abusive, disrespectful, normal and different combinations of each. The dataset presented [7] is in English and it also captures the annotators sentiments, directness, intended targets and groups. These attributes are important but to create a homogeneous dataset these have been left out from this study. The total size of this dataset is around 5647 instances which are reorganised for our purpose into three categories, hateful, abusive, or neither, with abusive content of around 26%.

SemEval 2019 Task 5: The dataset presented [8] comprises 12906 text instances of which 42% are abusive in nature. Again the data is sourced from twitter domain. This is a shared task with three sub-task in it. The first task is a binary task to classify if the text instance is hate or not, which is what we will consider for our study. The other tasks are within hate, if the text is directed towards a group

[1] www.nohatespeechmovement.org
[2] hatebase.org

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 25 November 2020 doi:10.20944/preprints202011.0646.v1

or individual and within hate, if the text is aggressive or not. This dataset is hate speech sequences against immigrants and women in English language.

PMathur et al: The dataset presented in the [9] contains tweets of offensive nature along with a profanity word list in code-mixed Hindi. This is a phonetic translation of a Hindi word written in English. These tweets are annotated into three different categories of normal, abusive and Hate.

For our study, the profanity word list has been updated with new words and with Hindi dialect references. Using this profanity word list, new tweets have been curated from twitter for both Hindi and code-mixed Hindi text. These new tweets have been manually annotated following the guidelines mentioned by the original author, details of which are given in Model Section. The resulting size of profanity word list is around 226 words in code mix Hindi, with the meaning of each in English, profanity score and devanagari script Hindi word. The size of the dataset is around 3189 with 55% tweets of abusive nature.

Table 2. Datasets Hate Composition

Dataset Name	*Total Records*	*Hate %*	*Abuse %*
HASOC2019 - EN	7005	36.38	-
HASOC2019 - HI	9330	52.92	-
TDavidson et al	24783	77.43	5.77
ElSherif et al	10760	90.94	-
Ousidhoum et al	5647	65.66	22.63
SemEval 2019 Task 5	12906	42.15	-
PMathur et al	3189	55.34	9.5
Total	**76403**	**61.84**	**4.55**

The data from all the sources have been collated and sorted into a homogeneous form, the details of which are illustrated in the Model section.

2.2. Related Models

Davidson et al. [5] in their work followed a very simple approach and created a very simple model. In this work, the author created simple feature vectors for each sample instance. The feature vector consisted of Part of Speech (POS), term frequency–inverse document frequency (TFIDF) vectors amongst others. Final model was a logistic regression and linear support vector machine (SVM) which claimed to achieve an overall precision 0.99, recall 0.90 and F1 score of 0.90.

In another work by [7] the author has created two types of models. Bag-of-word (BOW) a feature based logistic regression model and a deep learning based model containing Bi-directional Long Short Term Memory (BiLSTM) layer with one hidden layer. The author claims, due to the small size of the dataset, deep learning based models performed poorly. The author suggests to use Sluice network [11] as they are suitable for loosely related tasks such as the annotated aspects of the corpora. State-of-art performance scores are obtained using Sluice networks.

The author also claimed that for a single task single language the model achieved an accuracy of 94% , this is in contrast against our dataset. We found that the model only achieved 68% accuracy for English language. We could not use Hindi or code-mixed Hindi due to the fact that the embedding used by the author, Babylon multilingual word embeddings [12] and MUSE [13] were not compatible with our Hindi or code-mixed Hindi dataset.

One of the initial research on hate speech detection from Hindi-English tweets was done by [14]. The research was based on 4575 code-mix Hindi tweets. The author worked on features like character n-gram, emoticon count, word n-gram, punctuation count and applied dimension reduction techniques. An accuracy of 71.7% on SVM and 66.7% on Random forest was obtained.

The most relevant research done for code-mixed Hindi dataset was done by [9]. The author claimed that term frequency, inverse document frequency (TF-IDF) and BoW features with SVM model

 doi:10.20944/preprints202011.0646.v1

gave peak performance when compared with other configurations of baseline supervised classifiers. The other features such as Linguistic Inquiry and word count features (LIWC) [15], profanity vector [16] with glove and twitter embeddings gave best results against the baseline model as argued by the author. Also, a transfer learning based approach called as Multi Channel Transfer Learning based Model (MTLM) achieved on its best configuration a precision of 0.851, recall of 0.905 and f1 score of 0.893 . Similar scores have been obtained in our study as well.

[17] applied deep learning based approaches. In the experiment the author created sub-word level LSTM model and Hierarchical LSTM model with attention based phonemic sub-words. The architecture of these models are important research in this field as attention based models perform highly in comparison to basic deep learning based models. Hierarchical LSTM with attention achieved an accuracy of 66.6% while sub-word level LSTM model scored 69.8%.

In the work of [18] three deep learning models using domain specific word embeddings were created. These word embeddings are available and comprise 255,309 code-mix Hindi text which the author collected from Twitter. The word embeddings are trained using gensim's word2vec model and are used in the three deep learning models. For 1D-CNN an accuracy of 82.62% was achieved which was the highest against BiLSTM with 81.48% and LSTM with 80.21%.

In the above mentioned research we have seen that all the models were created to process only a single language at a time. In our project we have focused on creating a single model that will inculcate multiple languages at the same time leading to a more generalisable approach. The only exception of a multilingual model found in the literature is that by [19] where the authors focus on English and Chinese. However, the authors used third party translation APIs like Google translation API, to convert the raw text into English or Chinese.

In the following sections we propose a Logistic regression model and two Deep Learning based models for both English and Hindi (code-mixed Hindi) languages. The model architecture for all the languages is the same, however the feature building process is different for different languages.

3. Model

In this section we discuss the model building procedure. We are building a system that learns from the feedback, is multilingual and is trained on large hate sequence text. We also provide optimisation carried out and performance verification.

3.1. Data Preprocessing

Since the data in our use case is sourced from various datasets [4–9] it becomes essential to organise it into a homogeneous form. To achieve this, we consider the text and the class type of the existing datasets. In many datasets, as seen in 2, the data is classified as hate or not, and abuse class is not present. The original text is taken without any formatting. The class types (if necessary) are converted from original to either, hate, abusive or normal. In order to identify the source of the text, we have labeled each record instance with specific dataset identifiers.

Samples for Hindi and code-mixed Hindi are sourced from [9] and [4]. This dataset is relatively small in comparison to English language dataset. Thus, we have updated this data sample by adding new tweet samples to Hindi and code-mix Hindi dataset by following the below procedure.

1. We have added and updated the profane word list provided by [9] with more words in code-mix Hindi language.
2. We have used this profane word list and added corresponding Hindi devnagari scripted text against each code-mix Hindi profane word.
3. In order to assign classes to newly curated text from Twitter Public API, we have followed the below guideline.

    (a) Tweets involving sexist or racial slur to target a minority, these tweets may contain abusive word are annotated as Hate,

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 25 November 2020 doi:10.20944/preprints202011.0646.v1

(b) Tweets which represent undignified stereotypes are marked as Hate
(c) Tweets which contain problematic hashtags are marked as Hate.
(d) Tweets which contain only abusive words are tagged as Abusive.
(e) Other tweets are marked as normal.

Typically few instances of our seventy six thousand dataset looks something like in the table 3.

Table 3. Example Texts

Dataset Type	Example	Class Type
English-EN	I am literally too mad right now a ARAB won #MissAmerica	Hateful
English-EN	Black on the bus	Hateful
English-EN	I can not just sit up and HATE on another b** .. I got too much sh** going on!	Abusive
Hindi-HI	ये लिटन की गाँड में लिप्टन की चाय डालता हूँ अभी विथ गर्म केतली	Hateful
Hindi Code-Mixed	Main jutt Punjabi hoon aur paka N league. Madarchod Imran ki Punjab say nafrat clear hai.	Hateful

One of the major issues related to twitter text and to which other studies have indicated as the potential cause for degrade in performance is the small sequence of text followed by ever evolving use of unknown words such as slang and hashtags words. As seen in Table 1, the Max Seq. Len. denotes the maximum sequence length of the various language datasets, the sequence length for all the languages is very small (less than 100), but the vocabulary size (vocab. size) is way too high. [20] describes the limitations of unusually large vocabulary leading to poor performance.

In our research, when we performed exploratory data analysis (EDA), we too concluded that due to above mentioned issues, the performance of our classifier was getting affected. Thus during prepossessing of text we decided to use ekphrasis [21]. This preprocessor is used to in the following way,

- Normalise - url, email, percent, money, phone, user, time, date, number in twitter text
- Annotate - hashtag, allcaps, elongated, repeated, emphasis, censored, words in the text
- Word Segmentation
- Convert Hashtags to unpacked word list (if possible)
- Convert English contractions such as "can't", "we'll" into "cannot" and "we will"
- Tokenise the sentences into words
- Convert Emoticons and Slangs into actual expression/phrase.

The output from this pre-processing technique contains more information than previously studied research. Although, ekphrasis works really well for English language but it does not have a multilingual aspect built into it. Therefore, we are leveraging [22], Indic NLP and [23], NLTK library support for Hindi Language. During the EDA, we observed some issues related to unpacking of slang, emoticons and English contractions in ekphrasis and tokenisation issues in [22]. These are now being rectified by raising an issue on Github and by making an open source contribution to the original work.

 doi:10.20944/preprints202011.0646.v1

3.2. Model Building

In previous research, we saw that BoW or TFIDF based feature vectors along with other features tends to work best with Logistic regression based models and they generally give a fine baseline model performance. Similar approach has been carried out in our study.

We created a Logistic regression model as the baseline model. We applied an L2 penalty giving equal class weights to three classes. To comprehend the sheer volume of data and to make the model converge, the model's hyper-parameter, maximum iteration is set to 5000 iterations for English, 3000 for Hindi and code-mix Hindi. Other hyper-parameters were obtained in grid search, with 5 fold cross validation, and the performance scores are described in Table 4.

Table 4. Performance Comparison of f1 scores and accuracy between classes for Logistic regression model

Dataset Type	*f1-Neither*	*f1-Hate*	*f1-Abuse*	*Acc*
EN	0.68	0.47	0.80	0.687
HI	0.95	0.96	-	0.956
HI-Code Mix	0.84	0.62	0.91	0.855

After building a baseline model, we explore the possibility of building a Hierarchical Deep Neural Network by combining several Convolutional Neural Network (CNN) filters into Bi-directional Long Short Term Memory network. During the EDA we discovered valuable sequential information in twitter text for each class. When we apply a BiLSTM layer on top of a CNN layer, we are able to capture the sequential information as well as the low lying textual representation. Thus we improve the simple contextual BiLSTM classification with use of CNN layers. This model has the following architecture:

1. Word Embedding - to capture and convert text into sequences.
2. CNN - to capture low lying information using different filter size
3. Bidirectional LSTM - to capture contextual information of the sequence.
4. Fully-Connected output

This model is inspired by the work of [24], [25] and [26]. In these researches, with use of CNN, the character level property of the text is explored.

Figure 1 describes the architecture and different layers. The word embedding layer converts sparse representation of word sequences into dense vector representation. The 3 CNN layers consist of 3 parallel convolutional, 2 dimensional filters, of size 3, 4 and 5 with batch normalisation and max pooling layer. This CNN layer is time distributed over the LSTM layer, which captures the contextual information and finally the model is fed forwarded to a dense layer, where sigmoid activation layer performs the classification. This model is built in order to analyse the use of deep neural networks and, if there can be any improvement in the previous benchmarks with respect to models performance.

The above network is trained on all three datasets and after optimisations the results obtained are relatively similar to the Logistic regression model.

Table 5. Performance Comparison of f1 scores and accuracy between classes for CNN-LSTM model

Dataset Type	*f1-Neither*	*f1-Hate*	*f1-Abuse*	*Acc*
EN	0.74	0.63	0.72	0.78
HI	0.86	0.86	-	0.85
HI-Code Mix	0.83	0.62	0.70	0.86

Table 4 describes the performance for three language datasets. This model is trained on GPU thus the model convergence time is way quicker in comparison to the logistic regression model.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 25 November 2020 doi:10.20944/preprints202011.0646.v1

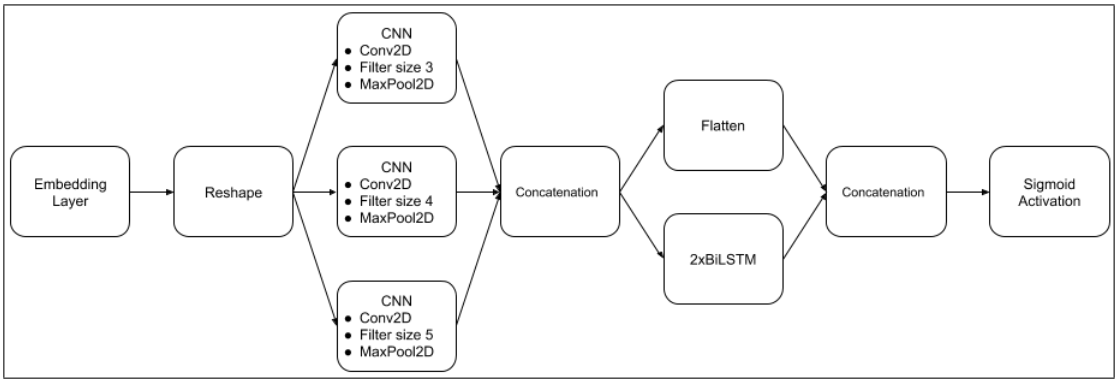

Figure 1. CNN LSTM Network.

We use random search to find the parameters and hyper-parameters. Table 5 describes the parameters which resulted in the best performance.

Table 6. Parameter Optimisation for CNN LSTM Model

Dataset Type	*epochs*	*batch* size	*optimiser*	*learning* rate	*dropout*	*hidden* size
EN	22	30	sgd	0.001	0.2	64
HI	20	30	sgd	0.001	0.1	32
HI-Code Mix	22	30	adam	0.01	0.2	64

In the previous model, the word embeddings are learned from data that have low dimension and are more dense than TFIDF features. But the model seemed to be less performant than the logistic regression model built previously. Therefore, in the next model we incorporate Bidirectional Encoder Representations from Transformers (BERT) [27] to leverage contextual word embeddings in place of CNN layers we previously added to the CNN LSTM model. This model has the following architecture:

1. Pre-trained BERT Embedding - to capture contextual word embeddings
2. Bidirectional LSTM - to capture contextual information of the sequence.
3. Fully-Connected output

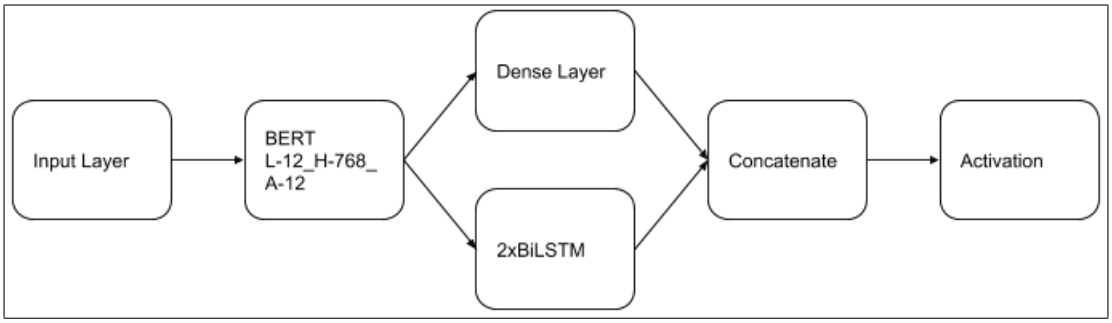

Figure 2. BERT based network.

In the work of [28], the author claims to achieve a best in class f1 and GLUE score of above 90% for SQUAD 1.1 dataset classification. Using transfer learning method, we leverage the pre-trained embeddings of this Transformer model into our CNN LSTM model. There are multiple pre-trained embeddings of various sizes (layers, hidden nodes) and languages. In our study, for English and Code Mix Hindi dataset we are using, BERT-Base, Uncased: 12-layer, 768-hidden, 12-heads, 110M parameters and for Hindi dataset we are employing BERT-Base, Multilingual Cased: 12-layer, 768-hidden, 12-heads, 110M parameters.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 25 November 2020 doi:10.20944/preprints202011.0646.v1

Figure 2 describes the architecture and different layers. Instead of word embedding and CNN layers, for English we use 24-layer contextual aware BERT language embeddings. There are only 12 hidden layers in the pre-trained model for BERT to support multilingual aspects which we are implementing for our Hindi dataset. The BERT model results in 768 hidden Layers which are then time distributed over the BiLSTM layers and finally concatenated on a dense layer. For classification, the sigmoid activation layer is applied to the dense layer.

Table 7. Performance Comparison of f1 scores and accuracy between classes for BERT model

Dataset Type	*f1-Neither*	*f1-Hate*	*f1-Abuse*	*Acc*
EN	0.75	0.55	0.90	0.80
HI	0.94	0.94	-	0.95
HI-Code Mix	0.83	0.65	0.92	0.86

Table 7 describes the performance score for the BERT based model. The model is trained on Google's Tensor Processing Unit (TPU) and requires different mechanisms to process the data and feed the data into the network. The model's performance is better than the Logistic regression model and CNN LSTM model.

Table 8. Parameter Optimisation for BERT Model

Dataset Type	*epochs*	*batch size*	*optimiser*	*learning rate*	*dropout*	*hidden size*
EN	13	121	sgd	0.01	0.2	3
HI	15	9	sgd	0.01	0.2	3
HI-Code Mix	8	11	sgd	0.01	0.2	3

In the table 8 we describe the best parameters obtained using the technique of random search and evaluation.

The models developed in this study are oriented to be used in an online environment. Therefore, it becomes important that we pursue a state of art model which can be re-trained in a given frame of time within a resource constrained environment. It is critical to analyse the prediction time as well as the time taken by the model to train.

Table 9. Model Evaluation and Performance Scores

Model Type	*Preprocess* Time	*Training* Time	*Infer.* Time	*System* Req.	*Model* size	*Test* Acc.
LR-EN	126sec	9287sec	0.7sec	CPU	498.8KB	68%
LR-HI	14927sec	412sec	0.7sec	CPU	161.2KB	95%
LR-HI Code-Mix	435sec	210sec	0.7sec	CPU	402.2KB	85%
CNN-LSTM-EN	41sec	340sec	13sec	GPU	19.9MB	78%
CNN-LSTM-HI	1453sec	68sec	11sec	GPU	5.7MB	85%
CNN-LSTM HI-Code-Mix	0.6sec	27sec	13sec	GPU	6.8MB	83%
BERT-EN	28sec	726sec	0.9sec	TPU	438.6MB	80%
BERT-HI	0.8sec	1037sec	0.9sec	TPU	712.2MB	95%
BERT-HI Code-Mix	0.1sec	218sec	0.9sec	TPU	438.6MB	86%

The table 9 summaries some of the important performance features of the models built so far.

 doi:10.20944/preprints202011.0646.v1

- The Logistic regression model takes an excessive amount of time for preprocessing and training. The BERT and CNN LSTM model takes much less time in comparison. This is due to the fact that the Logistic regression model requires TFIDF features on the entire vocabulary size.
- The model size for Logistic regression models are thousand times less than the BERT model and 100 times less than CNN LSTM model. Thus the inference time for the Logistic model is the least as compared to other models.
- The logistic regression model is well suited for smaller datasets as it is trained on CPU, for larger datasets GPU or TPU based CNN LSTM or BERT based models could be used to reduce the training and inference time.
- One of the reasons for high inference time for CNN LSTM models is inability to find the right set of hyper-parameters, and so we see a higher performance for BERT models as compared with other models.

4. Results and Discussion

We show that the Logistic regression model supplemented with TFIDF and POS features gave relatively good results in comparison to other models but the time taken for the model to converge is very long. These results presented are in line with previous research [5,9,29].

The Deep learning model, gave similar performance scores without much feature engineering and the model converged quickly too. Such results can be attributed to use of embedding layers in the neural network. We also tried to use the glove and twitter embedding layers but due to the high number of unknown words, desirable results were not obtained.

The logistic regression model has less system requirements, is very quick to infer a single sample instance and has great performance scores. Due its inability to scale on a large dataset and the time taken to build the model is very high. It can only be used as a benchmark for other models and cannot be used in an online environment, where models are continuously re-trained on the feedback loop.

The CNN LSTM model is an average model which has a mediocre performance but it's performance can be perfected. The random search optimization used in this study is a test driven approach. Other optimization techniques like bayesian optimization can find better hyper-parameters. But the time taken to build the model will increase considerably. The CNN LSTM model requires a moderate GPU processing and infer a single sample instance in near-real time. Thus we are utilizing this model in our online web application.

The BERT model is a high performance state of the art model. It has the highest accuracy amongst all the models. If the system requirement(TPU) criteria is fulfilled, it can be used in an online environment. This model has very less training time and is very quick to infer a single sample instance. The only constraint is the use of the Tensor Processing Unit. Though the model size is 1000 times the size of a logistics regression model, memory and disk space consumption is hardly a worry these days due to their easy availability. Unlike GPU the cost of a single TPU instance is very high. Still the scope of TPU usage as the main stream processing unit for machine learning in near future is high. Thus, the research done in this study is futuristic and this novel state-of-the-art model is very much applicable to be used in the real world.

In order to give a fair comparison of our models to existing ones, we apply our models to the datasets in isolation, we find that our models outperforms on most of the datasets. Table 10 shows these results. The dataset was segmented in the exact proportions of test and train as it was done in the original research. Further, out of the many models we described here, we have chosen the best results (after optimising it to work on smaller datasets) of CNN LSTM model as it works faster in loading the data as well as giving inference. The results are consistent with the performance which is consistent as well across the datasets.

Further, we discuss some categories of errors that were observed in the deep learning and logistic regression models:

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 25 November 2020 doi:10.20944/preprints202011.0646.v1

Table 10. Comparison of different dataset models with our model

Datasets	Existing Best Scores			Our Model Best Scores		
	Precision	*Recall*	*F1*	*Precision*	*Recall*	*F1*
HASOC 2019 EN	n.a.	n.a.	0.79	0.91	0.9	**0.9**
HASOC 2019 HI	n.a.	n.a.	**0.81**	0.87	0.82	**0.81**
Davidson et al. 2017	0.91	0.9	0.9	0.93	0.9	**0.92**
ElSherif et al	n.a.	n.a.	n.a.	0.85	0.86	**0.83**
Ousidhoum et al. 2019	n.a.	n.a.	**0.94**	0.91	0.91	0.9
SemEval 2019 Task 5	0.69	0.68	0.65	0.83	0.8	**0.8**
Mathur et al. 2018.	0.85	0.9	**0.89**	0.88	0.9	0.87

- The noisy and repetitive nature of the data present on social media creates a skewness in the class distribution. Although it is taken care of by hyper-parameter tuning, nonetheless, it still caused overfitting in both logistic regression models and CNN LSTM models.
- The code mixed words tend to be biased as they appear at specific locations. [30] showed that bilingual languages favor code mixing of specific words at specific locations.
- Almost all the class labels are hand annotated. Since there is no defined criteria of how one should classify, it can lead to ambiguity between classes. This is the reason for lower f1 scores between abusive and hate classes in both logistic regression and CNN LSTM models for all the languages.

5. Online Feedback Mechanism

In this section, we discuss the use of created models in an online environment. Once the model is created it is essential to understand how the model will behave. Here, we will also discuss the complete execution of the machine learning pipeline; by adding an external feedback loop to the model. This allows the model to learn the evolution of text and textual context over time. We are aware that there are several disadvantages of doing this like the model may become biased towards one class if used over a stretch of time, but we can overcome this by adding human-in-the-loop. In this way the weight of tagging text by moderator and annotators could be reduced significantly.

To achieve this, we create a RESTful API which connects the machine learning model to an online web chat system. The web chat we create here is a live application demonstrating a chat room. The most important aspect is the API which scans the page and scores each comment by the user. It summarizes the total score of all the classes viz. normal, hateful or offensive. In the image 3 we see the metric which understands the page content with respect to it's hateful or abusive nature.

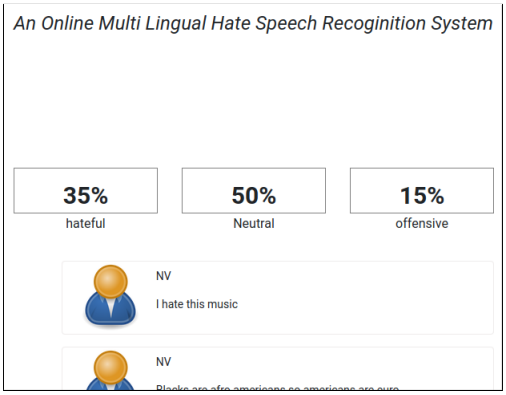

Figure 3. Real time monitoring of content and scoring page based on in percentage if Hateful content is present.

 doi:10.20944/preprints202011.0646.v1

The above use case is a passive method showcasing the capability of our model in an online application. An active use case would be if we can prevent the use of hateful/abusive comments. In order to do so, we create a notifier on the submit button of the comments. If the comment is hateful or abusive, the user gets notified and we can actively prevent users from commenting derogatory remarks. Thus, users can be made self aware and can be advised to refrain from sending something incinerating to the social world. Figure 4 depicts the mentioned use case.

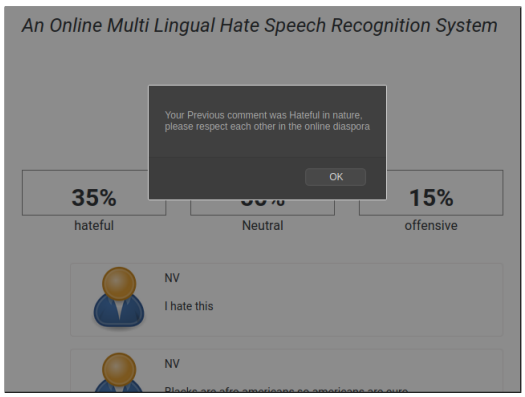

Figure 4. Proactively notifying user that their comment is hateful in nature.

The above application has the capability to automatically switch between different machine learning models depending on the language of the user. If the user uses Hindi, the model automatically switches to the Hindi model.

We capture these online comments into a database and use this database to update our existing datasets. The database can be reviewed by an annotator for comments which have lower confidence of belonging to a single class. Thus a very small portion of the comments have to be reviewed. Such comments can be added to the training dataset for future addition.

6. Conclusion

The objective of this study was to bring to light a model which was trained on a large dataset of multilingual languages. By experimenting on an aggregated dataset combining six languages in English, Hindi and Code-mixed Hindi, we demonstrate that our models achieve comparable or superior performance to a wide range of baseline monolingual models. The model leads to competitive performance on combined data and works in an online environment in near-real time.

Further work can be done both in the space of improving the model architecture. We can add more use cases to the online system. From the Logistic Regression point of view, we can apply feature selection methods to choose most prominent features and extend the model to other code-mixed languages. We can exploit other advanced hierarchical models such as Spinal, GRU. Bayesian optimization and other fine tuning deep learning models and hyper-parameter selection are some of the areas of future research.

7. References

1. Cortese, A.J.P. *Opposing hate speech*; Greenwood Publishing Group, 2006.
2. Abusive and Offensive Online Communications: A Scoping Report, 2018.
3. Vidgen, B.; Derczynski, L. Directions in Abusive Language Training Data: Garbage In, Garbage Out. *arXiv preprint arXiv:2004.01670* **2020**.
4. Mandl, T.; Modha, S.; Majumder, P.; Patel, D.; Dave, M.; Mandlia, C.; Patel, A. Overview of the HASOC Track at FIRE 2019: Hate Speech and Offensive Content Identification in Indo-European Languages.

Proceedings of the 11th Forum for Information Retrieval Evaluation; Association for Computing Machinery: New York, NY, USA, 2019; FIRE '19, p. 14–17. doi:10.1145/3368567.3368584.

5. Davidson, T.; Warmsley, D.; Macy, M.; Weber, I. Automated Hate Speech Detection and the Problem of Offensive Language, 2017, [arXiv:cs.CL/1703.04009].
6. ElSherief, M.; Nilizadeh, S.; Nguyen, D.; Vigna, G.; Belding, E. Peer to Peer Hate: Hate Speech Instigators and Their Targets, 2018, [arXiv:cs.SI/1804.04649].
7. Ousidhoum, N.; Lin, Z.; Zhang, H.; Song, Y.; Yeung, D.Y. Multilingual and Multi-Aspect Hate Speech Analysis. Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP); Association for Computational Linguistics: Hong Kong, China, 2019; pp. 4675–4684. doi:10.18653/v1/D19-1474.
8. Basile, V.; Bosco, C.; Fersini, E.; Nozza, D.; Patti, V.; Rangel Pardo, F.M.; Rosso, P.; Sanguinetti, M. SemEval-2019 Task 5: Multilingual Detection of Hate Speech Against Immigrants and Women in Twitter. Proceedings of the 13th International Workshop on Semantic Evaluation; Association for Computational Linguistics: Minneapolis, Minnesota, USA, 2019; pp. 54–63. doi:10.18653/v1/S19-2007.
9. Mathur, P.; Sawhney, R.; Ayyar, M.; Shah, R. Did you offend me? Classification of Offensive Tweets in Hinglish Language. Proceedings of the 2nd Workshop on Abusive Language Online (ALW2); Association for Computational Linguistics: Brussels, Belgium, 2018; pp. 138–148. doi:10.18653/v1/W18-5118.
10. Waseem, Z.; Hovy, D. Hateful Symbols or Hateful People? Predictive Features for Hate Speech Detection on Twitter. Proceedings of the NAACL Student Research Workshop; Association for Computational Linguistics: San Diego, California, 2016; pp. 88–93.
11. Ruder, S.; Bingel, J.; Augenstein, I.; Søgaard, A. Sluice networks: Learning what to share between loosely related tasks. *ArXiv* **2017**, *abs/1705.08142*.
12. Smith, S.L.; Turban, D.H.P.; Hamblin, S.; Hammerla, N.Y. Offline bilingual word vectors, orthogonal transformations and the inverted softmax, 2017, [arXiv:cs.CL/1702.03859].
13. Lample, G.; Conneau, A.; Denoyer, L.; Ranzato, M. Unsupervised Machine Translation Using Monolingual Corpora Only, 2017, [arXiv:cs.CL/1711.00043].
14. Bohra, A.; Vijay, D.; Singh, V.; Akhtar, S.S.; Shrivastava, M. A Dataset of Hindi-English Code-Mixed Social Media Text for Hate Speech Detection. Proceedings of the Second Workshop on Computational Modeling of People's Opinions, Personality, and Emotions in Social Media; Association for Computational Linguistics: New Orleans, Louisiana, USA, 2018; pp. 36–41. doi:10.18653/v1/W18-1105.
15. Sawhney, R.; Manchanda, P.; Singh, R.; Aggarwal, S. A Computational Approach to Feature Extraction for Identification of Suicidal Ideation in Tweets. Proceedings of ACL 2018, Student Research Workshop; Association for Computational Linguistics: Melbourne, Australia, 2018; pp. 91–98. doi:10.18653/v1/P18-3013.
16. Jay, T.; Janschewitz, K. The pragmatics of swearing. *Journal of Politeness Research-language Behaviour Culture - J POLITENESS RES-LANG BEH CUL* **2008**, *4*, 267–288. doi:10.1515/JPLR.2008.013.
17. Santosh, T.; Aravind, K. Hate Speech Detection in Hindi-English Code-Mixed Social Media Text. 2019, pp. 310–313. doi:10.1145/3297001.3297048.
18. Kamble, S.; Joshi, A. Hate Speech Detection from Code-mixed Hindi-English Tweets Using Deep Learning Models, 2018, [arXiv:cs.CL/1811.05145].
19. Sohn, H.; Lee, H. MC-BERT4HATE: Hate Speech Detection using Multi-channel BERT for Different Languages and Translations. 2019 International Conference on Data Mining Workshops (ICDMW), 2019, pp. 551–559.
20. Chen, W.; Su, Y.; Shen, Y.; Chen, Z.; Yan, X.; Wang, W.Y. How Large a Vocabulary Does Text Classification Need? A Variational Approach to Vocabulary Selection. Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers); Association for Computational Linguistics: Minneapolis, Minnesota, 2019; pp. 3487–3497. doi:10.18653/v1/N19-1352.
21. Baziotis, C.; Pelekis, N.; Doulkeridis, C. DataStories at SemEval-2017 Task 4: Deep LSTM with Attention for Message-level and Topic-based Sentiment Analysis. Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017); Association for Computational Linguistics: Vancouver, Canada, 2017; pp. 747–754.

22. Kunchukuttan, A. The IndicNLP Library. https://github.com/anoopkunchukuttan/indic_nlp_library/blob/master/docs/indicnlp.pdf, 2020.
23. Loper, E.; Bird, S. NLTK: The Natural Language Toolkit. In Proceedings of the ACL Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics. Philadelphia: Association for Computational Linguistics, 2002.
24. Zhang, X.; LeCun, Y. Text Understanding from Scratch, 2015, [arXiv:cs.LG/1502.01710].
25. Jozefowicz, R.; Vinyals, O.; Schuster, M.; Shazeer, N.; Wu, Y. Exploring the Limits of Language Modeling, 2016, [arXiv:cs.CL/1602.02410].
26. Kim, Y.; Jernite, Y.; Sontag, D.; Rush, A.M. Character-Aware Neural Language Models, 2015, [arXiv:cs.CL/1508.06615].
27. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers); Association for Computational Linguistics: Minneapolis, Minnesota, 2019; pp. 4171–4186. doi:10.18653/v1/N19-1423.
28. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv preprint arXiv:1810.04805* **2018**.
29. Badjatiya, P.; Gupta, S.; Gupta, M.; Varma, V. Deep Learning for Hate Speech Detection in Tweets. *Proceedings of the 26th International Conference on World Wide Web Companion - WWW '17 Companion* **2017**. doi:10.1145/3041021.3054223.
30. Singh, R. Grammatical Constraints on Code-Mixing: Evidence from Hindi-English. *Canadian Journal of Linguistics/Revue canadienne de linguistique* **1985**, *30*, 33–45. doi:10.1017/S0008413100010677.