Cyberbullying Tweet Classification and Sentiment Analysis

Vishal Bansal
vishal19217@iiitd.ac.in

Apoorv Singh
apoorv19151@iiitd.ac.in

Sagar Suman
sagar19197@iiitd.ac.in

1. Introduction

Cyber Bullying is a term often used for when someone harasses or bullies others on the internet and other digital spaces specially social media sites. It affects the victims mental health drastically.So it motivates us to make a model which will automatically detect the tweet sentiment and if a tweet is found to be of bullying nature then the site can automatically remove it from its database. So our model task is to classify the sentiment of a tweet into 6 classes: religion, ethnicity, age, gender, not bullying, other_cyber bullying which then can be used by site.

2. Related Work

We gained an overview of the domain of text classification through this paper [green5]. Apart from this brief overview, we explored multiple research papers which employ different embeddings for machine learning and deep learning algorithms. We also noted that many papers are using Naive Bayes as their baseline model [green3] because of its simplicity and effectiveness. In Toxic Comment Detection Using LSTM [green1], they have used SpaCy word embeddings and they obtained an accuracy of 94% using LSTM. In the paper by Khambe and Joshi [green2] on Hindi-English mixed hate speech classification, they have used CNN-1D model and gained precision, recall and F1-score as 83.34, 78.51 and 80.85 respectively. Another study was conducted for sentimental analysis using BERT Embedding [green4] and they achieved 94% accuracy in BERT Small and 94.7% in BERT large architecture.

3. Dataset and Evaluation

The corpus consists of more than 47000 tweets of 6 classes. It is split into 90% training and 10% testing set. 90% is further reduced into 5 folds to tune the hyperparameters using results obtained from the validation set.Using Scatter plot of dataset we found that the dataset had a slight imbalance for other_cyberbullying class due to its less samples as compared to other classes. The size of Majority tweets were in range 2-40 words.

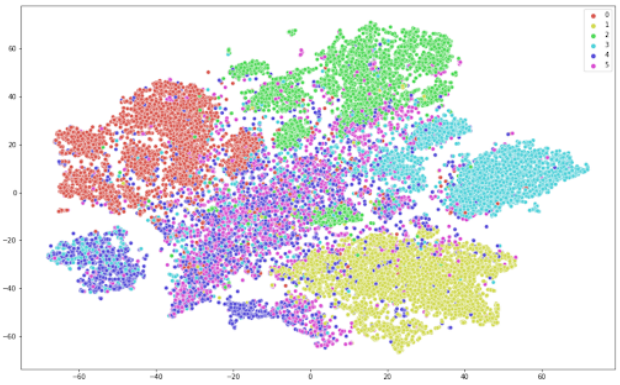

Figure 1. TSNE Scatter plot showing class separability

3.1. Feature Engineering

- Removed the duplicate tweets.
- Several NLP techniques such as removing stopwords , punctuations, emojis,HTML tags,URLs,spelling correction,special characters etc were used.
- Stemmization on tweets was performed.
- Analysis on Tweet lengths suggested us to remove tweets of length < 2 and length >100.
- Now Several word embedding techniques(Bag of Words,TF-IDF,Word2Vec,Glove) were incorporated and comparative study was done to check the most efficient embedding technique on several baseline Machine Learning Models(Gaussian Naive Bayes and Logistic Regression).

3.2. Evaluation Metrics

Classification Report comprises Accuracy,Precision,Recall,F1 score was used for Comparative Analysis for different models on Training and Testing Sets.
TP = True Positive
FP = False Positive
FN = False Negative

TN = True Negative

$$Precision = \frac{TP}{FP + TP}$$
$$Recall = \frac{TP}{FN + TP}$$
$$F1Score = \frac{2 * precision * recall}{precision + recall}$$
$$Accuracy = \frac{TP + TN}{TP + FN + FP + TP}$$

4. Methodolgy

4.1. Preprocessing :

After preprocessing using the steps mentioned in feature engineering we got cleaned tweets :-

	Basic clean	sentiment	tweet_length	Stemming	Lemmatization	Lemma_Stemming
0	words katandandre food crapilicious mkr	not_cyberbullying	5	word katandandr food crapilici mkr	word katandandre food crapilicious mkr	word katandandr food crapilici mkr
1	aussietv white mkr theblock today sunrise stud...	not_cyberbullying	10	aussietv white mkr theblock today sunris studi...	aussietv white mkr theblock today sunrise stud...	aussietv white mkr theblock today sunris studi...
2	classy whore red velvet cupcakes	not_cyberbullying	5	classi whore red velvet cupcak	classy whore red velvet cupcake	classi whore red velvet cupcak
3	meh p thanks heads concerned another angry dud...	not_cyberbullying	9	meh p thank head concern anoth angri dude twitter	meh p thanks head concerned another angry dude...	meh p thank head concern anoth angri dude twitter
4	isis account pretending kurdish account like i...	not_cyberbullying	8	isi account pretend kurdish account like islam...	isi account pretending kurdish account like is...	isi account pretend kurdish account like islam...

Figure 2. Preprocessed Tweets

4.2. Class Imbalance:

We found that our dataset was imbalance due to less samples of other cyberbullying class.To fix that we used the Random Oversampling method which sample equal number of dataset from each class.

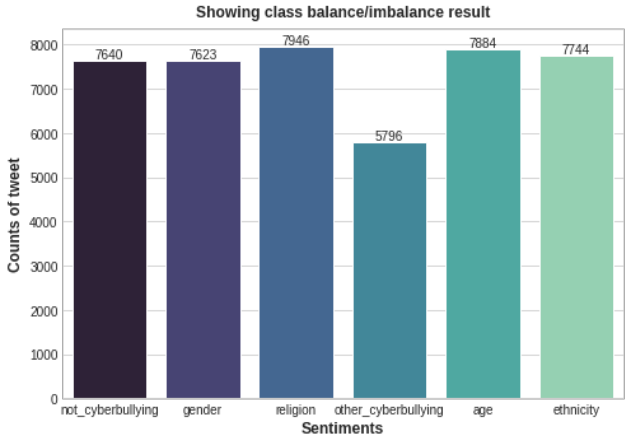

Figure 3. Class Imbalance

4.3. Baseline Model:

1. Logistic Regression:- used for training and testing the model performance.Hyperparameters used were 10000 for max iteration and since its was multiclass classification so 'ovr' (OnevsRest) method was used. Model was feed with features extracted from the tweets using different word embeddings. We plotted learning curves and used K-Fold validation to check whether our model is underfitting/overfitting for each embedding.

   - With TF-IDF word embedding:- So model was trained perfectly.
   - With Bag of Words word embedding:- Slight overfitting was observed in bag of word.
   - With Glove word embedding:- Model has trained perfectly
   - With Word2Vec word embedding:- Model has trained perfectly

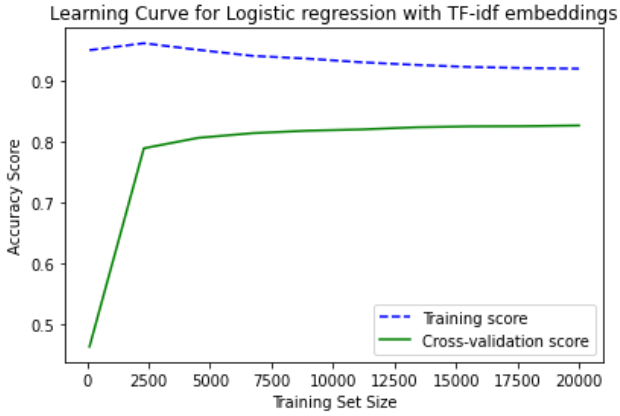

Figure 4. Learning Curve using TF-IDF word Embedding

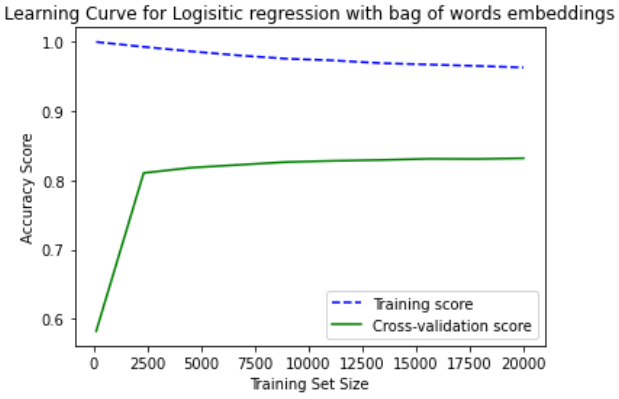

Figure 5. Learning Curve using Bag of word word Embedding

2. Naive Bayes:- used for training and testing the model

   - With TF-IDF word embedding:- So model was trained perfectly.

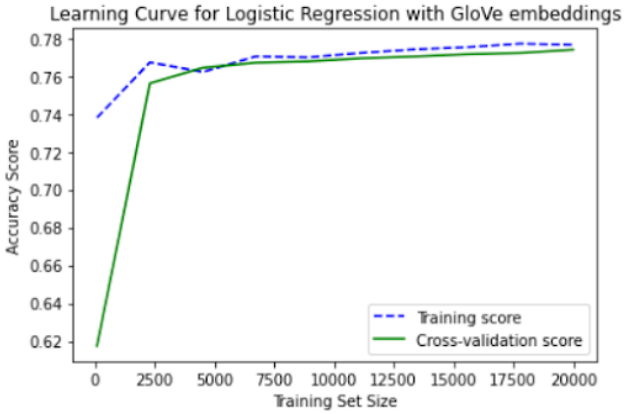

Figure 6. Learning Curve using Glove word Embedding

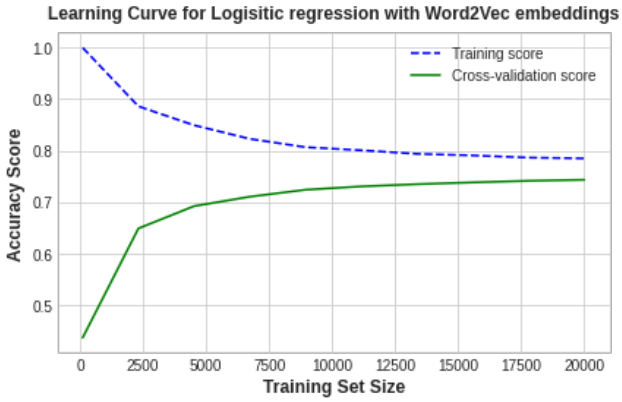

Figure 7. Learning Curve using Word2Vec word Embedding

- With Bag of Words word embedding:- Slight overfitting was observed in bag of word.
- With Glove word embedding:- Model has trained perfectly
- With Word2Vec word embedding:- Model has Slight Overfitting perfectly

.

4.4. Advanced Model

1. BERT(Base Model):- It is transformer based model which use several encoding layers by stack them and itself generate embedding for the dataset and hence generate the feature vector that can be further be feed to any classifier.Input is converted to Masked Sentences and Postion Embeddings using mask Language Model .Then tweets relations are learned using next sentence prediction model which composed of neural network layers also known as Attention and self Attention part.We have used the Pre trained BERT Model on HuggingFace Dataset and fine tune it with our dataset of 6 classes by adding a sequential layer comprising softmax layer for classifying 6 classes. Adam Optimiser and cross entropy loss was used since using adam we can optimise our learning rate well.

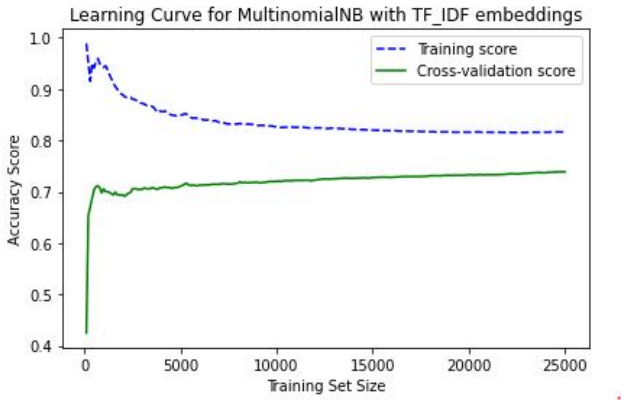

Figure 8. Learning Curve using TF-IDF word Embedding

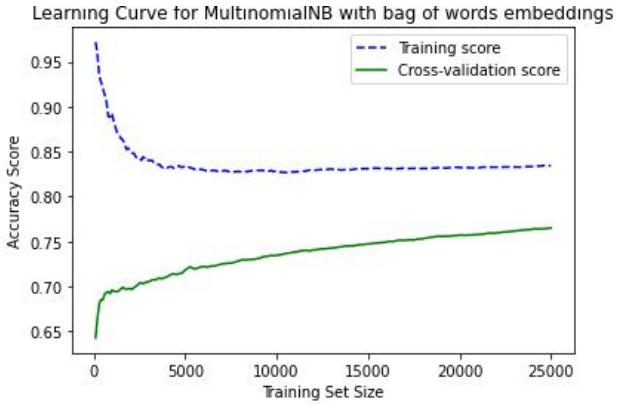

Figure 9. Learning Curve using Bag of word word Embedding

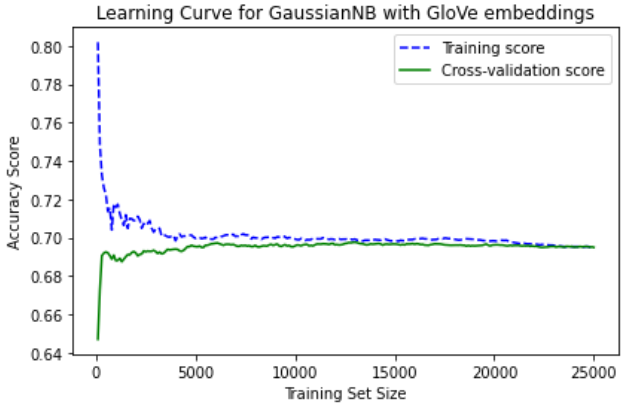

Figure 10. Learning Curve using Glove word Embedding

We observe that the training error is decreasing with

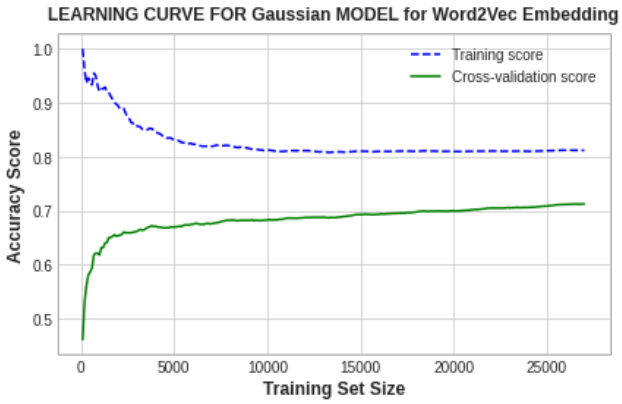

Figure 11. Learning Curve using Word2Vec word Embedding

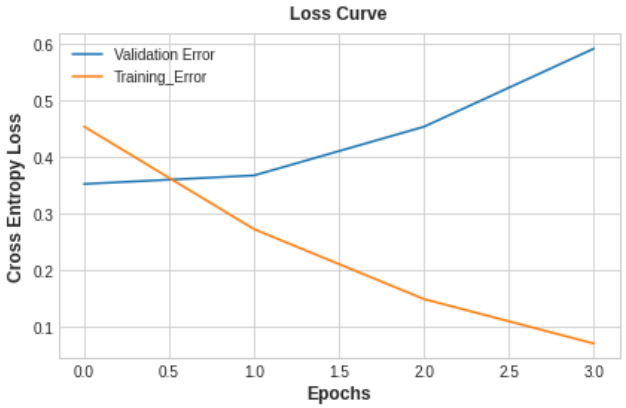

Figure 12. Loss Curve for BERT

each epochs but the validation error is increasing since with each epochs validation error increases.

2. LSTM:-LSTM was used with GloVe and Word2Vec embeddings to classify tweets. Since the input to a neural network is of fixed size, each tweet was tokenized and padded with zeros so that they have a length of 100. The input layer of the neural network contained 100 neurons. These neurons were connected to embedding layer. The embedding layer consists of 45551 words and each word has an embedding of size 200 for both GloVe and Word2Vec. The output of the embedding layer is of size 200 neurons which goes into the LSTM layer. The LSTM has only a single layer which is bidirectional and the size of the input to this layer is 200. After LSTM, there is a linear layer that has 200 input features and output has 6 features on which the softmax function is applied to get the classification. Here negative of log-likelihood loss was used which is useful to train a classification problem with a given no of classes. Optimizer used was AdamW.
We observe that the model is perfectly trained and

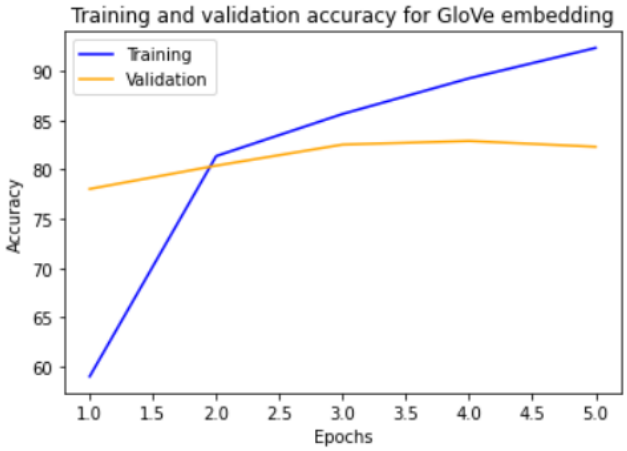

Figure 13. Accuracy curve for LSTM with Glove

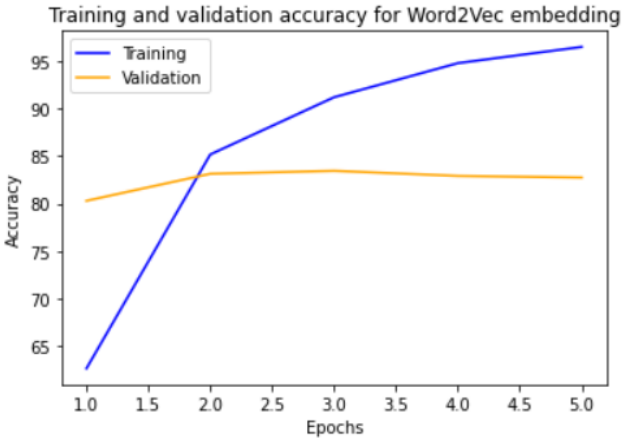

Figure 14. Accuracy curve for LSTM with Word2Vec

training and validation accuracy is increasing with each epochs.Though after some epochs the validation accuracy is about to decrease slightly as with increasing epochs model will slightly start to overfit.

3. CNN 1-D:- We have implemented a One dimensional Convolution network for the four word embeddings - Bag of words, Tf-idf, GloVe and Word to vector. The network takes 500 dimensional word embeddings as input. This 500 dimensional input is then passed to the first convolution layer with filter size as 2 and 32 feature maps. We have used padding to ensure boundary conditions and accurate analysis. Followed by this we have a max pooling layer which will help to reduce the size of feature maps, enhancing speed and help in reducing overfitting. Similar to this we have added 4 more layers which contain convolution followed by max pooling. After the end of layer eight (including input layer) we flatten the layer which contains 128 neurons. Before passing these 128 neurons to the

dense layer we used dropout rate to be 0.25 to reduce overfitting. Followed by this dropout we have another dense layer of 32 neurons. Finally, by again using the dropout rate to be 0.25 we reach the output layer. We have come up with this architecture by trying out multiple architectures, changing filter sizes and changing max pooling layer sizes. We have used the ReLu activation function between the layers. We choose this activation function over sigmoid or tanh because of its fast speed and resistance towards vanishing gradient..For loss function we have chosen categorical cross entropy loss function .Also,our last layer uses softmax as activation function. And this softmax activation function will prevent loss (cross-entropy loss) to shoot up to nan values.

For the output layer, we will get the probability (output from softmax) for each of the six classes. These probabilities represent how much the given tweet is likely to belong to a particular class. To get the output (predicted) label, we are using argmax over these probabilities.

We observed that except Bag of Words all other embeddings trained well on using 1-D CNN. .

5. Result and Analysis

The test accuracy we found for various baseline model and embedding are:-

Embeddings	Naive Bayes	Logistic Regression
Bag of Words	76%	84%
TF-IDF	74%	83%
Glove	70%	77%
Word2Vec	43%	75%

We found that Logistic Regression performed much better than Naive Bayes.The reason we analyzed was since Naive Bayes was a generative model and Logistic Regression was a discriminative model. So with increase in data points, discriminative models performed better than generative models.

Also, Naive Bayes assumes that the features are conditionally independent. But real data sets are never perfectly independent but they can be close. But Logistic Regression makes prediction for the probability using a direct functional form.

We can also observe that Word2Vec and Glove have less accuracy as compared with TF-IDF and Bag of Words. It is due to fact that Word2Vec and Glove are mostly use for deep learning models their they outperform the TF-IDF and Bag of Words. But TF-IDF,Bag of Words are mainly suitable in machine learning models.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv1d (Conv1D)             (None, 500, 32)           96

 max_pooling1d (MaxPooling1D  (None, 100, 32)          0
 )

 conv1d_1 (Conv1D)           (None, 100, 64)           6208

 max_pooling1d_1 (MaxPooling  (None, 20, 64)           0
 1D)

 conv1d_2 (Conv1D)           (None, 20, 64)            16448

 max_pooling1d_2 (MaxPooling  (None, 2, 64)            0
 1D)

 flatten (Flatten)           (None, 128)               0

 dropout (Dropout)           (None, 128)               0

 dense (Dense)               (None, 32)                4128

 dropout_1 (Dropout)         (None, 32)                0

 dense_1 (Dense)             (None, 6)                 198

=================================================================
Total params: 27,078
Trainable params: 27,078
Non-trainable params: 0
_________________________________________________________________
```

Figure 15. 1-D CNN Architecture

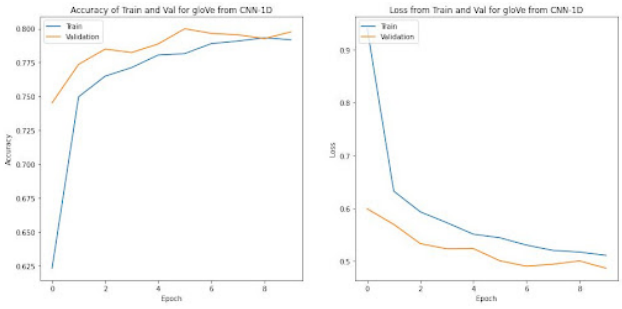

Figure 16. Accuracy curve for CNN-1D with Glove

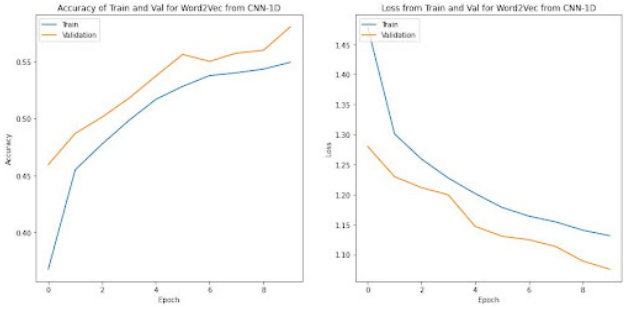

Figure 17. Accuracy curve for CNN-1D with Word2Vec

The test accuracy we found for various advanced models and embeddings are:-

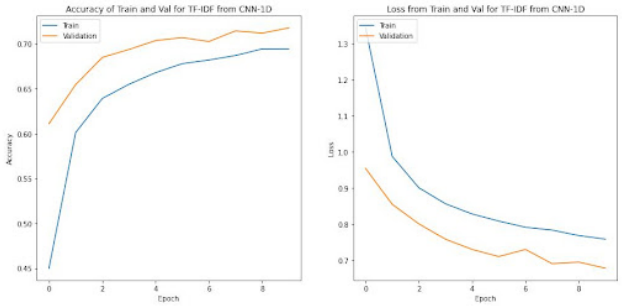

Figure 18. Accuracy curve for CNN-1D with TF-IDF

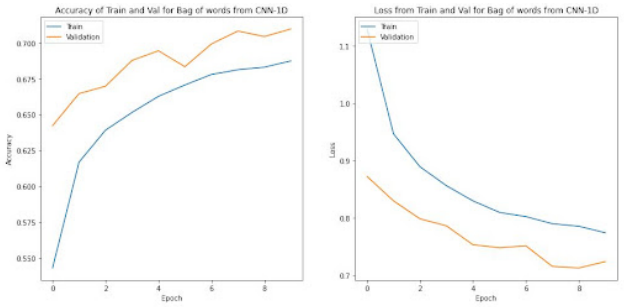

Figure 19. Accuracy curve for CNN-1D with Bag Of Word

Model(with Embedding)	Accuracy
BERT(base)	86%
LSTM(glove)	83%
LSTM(word2vec)	82%
CNN(BagOfWord)	70.4%
CNN(glove)	79.71%
CNN(word2vec)	57.97%

Here We can clearly observe that LSTM and BERT performed much better since they are deep neural network models whose accuracies increases with added layers and larege dataset.Also CNN with 1-D Also Performed well on Glove embedding.Although we can see that the accuracies are below 90 % for all models. For 1 model it is below 60 %. These all limitations are due the fact that the dataset has Other _ cyberbullying class which was overlapping with other classes so it was degrading the separation process by including itself as noise. Hence this was the only limitation we observed,

6. Contributions

6.1. Deliverable

- Apoorv has done the LSTM model and for baseline he implemented the baseline models using glove embeddings.
- Vishal has done the BERT model and for baseline he implemented the baseline models using word2Vec embeddings.
- Sagar has done the 1-D CNN model and for baseline he implemented the baseline models using Bag of Words and TF-IDF embeddings..

6.2. Citations

- Cyberbullying tweet classification Kaggle links:

magentahttps : / / www . kaggle . com / ludovicocuoghi / detecting - bullying - tweets-w-pytorch-bi-lstm

- For metrics: magentahttps://ml-explained.com/blog/metrics-explained
- For LSTM magentahttps : / / towardsdatascience . com / lstm - networks - a - detailed - explanation - 8fae6aefc7f9
- For BERT: magentahttps : / / towardsdatascience . com / bert - explained - state - of - the - art - language-model-for-nlp-f8b21a9b6270
- For 1- CNN magentahttps : / / www . tutorialspoint.com/tensorflow/index.htm
- For Pytorch with LSTM magentahttps : //pytorch.org/docs/stable/generated/torch.nn.LSTM.html
- For Baseline Models magentahttps://cs229.stanford.edu/proj2019spr/report/69.pdf

Thank you.

6.3. Github Repo Link

magentahttps://github.com/vishal19217/CyberBullying_Sentiment_Analysis

7. References

References

[1] Krishna Dubey, Rahul Nair, Mohd. Usman Khan, and Prof. Sanober Shaikh. Toxic comment detection using lstm. In *2020 Third International Conference on Advances in Electronics, Computers and Communications (ICAECC)*, pages 1–8, 2020. red1

[2] Satyajit Kamble and Aditya Joshi. Hate speech detection from code-mixed hindi-english tweets using deep learning models. *CoRR*, abs/1811.05145, 2018. red1

[3] Sang Bum Kim, Hae-Chang Rim, Dongsuk Yook, and Heui Seok Lim. Effective methods for improving naive bayes text classifiers. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 2417 of *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, pages 414–423. Springer Verlag, 2002. 7th Pacific Rim International Conference on Artificial Intelligence, PRICAI 2002 ; Conference date: 18-08-2002 Through 22-08-2002. red1

[4] Manish Munikar, Sushil Shakya, and Aakash Shrestha. Fine-grained sentiment classification using bert. In *2019 Artificial Intelligence for Transforming Business and Society (AITB)*, volume 1, pages 1–5, 2019. red1

[5] Fabrizio Sebastiani. Machine learning in automated text categorization. *CoRR*, cs.IR/0110053, 2001. red1