Application of Natural Language Processing in Detecting Cyberbullying

A thesis Report
Submitted in partial fulfillment of the requirements for the Degree of
Bachelor of Science in Computer Science and Engineering

Submitted by

Grownthona Rahman	**190104079**
Sean Rahman	**190104103**
Rifat Bin Karim	**190104105**
Dewan Mubinul Haque	**190104150**

Supervised by

Ms. Syeda Shabnam Hasan

Department of Computer Science and Engineering
Ahsanullah University of Science and Technology

Dhaka, Bangladesh

November 21, 2023

CANDIDATES' DECLARATION

We, hereby, declare that the thesis presented in this report is the outcome of the investigation performed by us under the supervision of Ms. Syeda Shabnam Hasan, Department of Computer Science and Engineering, Ahsanullah University of Science and Technology, Dhaka, Bangladesh. The work was spread over two final year courses, CSE4100: Project and Thesis I and CSE4250: Project and Thesis II, in accordance with the course curriculum of the Department for the Bachelor of Science in Computer Science and Engineering program.

It is also declared that neither this thesis nor any part thereof has been submitted anywhere else for the award of any degree, diploma or other qualifications.

Grownthona Rahman
190104079

Sean Rahman
190104103

Rifat Bin Karim
190104105

Dewan Mubinul Haque
190104150

CERTIFICATION

This thesis titled, **"Application of Natural Language Processing in Detecting Cyberbullying"**, submitted by the group as mentioned below has been accepted as satisfactory in partial fulfillment of the requirements for the degree B.Sc. in Computer Science and Engineering in November 21, 2023.

Group Members:

Grownthona Rahman	**190104079**
Sean Rahman	**190104103**
Rifat Bin Karim	**190104105**
Dewan Mubinul Haque	**190104150**

Ms. Syeda Shabnam Hasan
Assistant Professor & Supervisor
Department of Computer Science and Engineering
Ahsanullah University of Science and Technology

Prof. Dr. Md. Shariar Mahubub
Professor & Head
Department of Computer Science and Engineering
Ahsanullah University of Science and Technology

ACKNOWLEDGEMENT

Let us begin by expressing our sincere gratitude and praises to the Almighty Allah for giving us the health and well-being necessary to finish our academic thesis.

We would first and foremost like to thank our parents for their blessings, which have allowed us to progress this far in life. Above all, we would like to express our sincere gratitude to our supervisor, Assistant Professor, Ms. Syeda Shabnam Hasan, who helped us throughout this journey. Her unwavering patience, continuous support, supervision, and immense knowledge enabled us to successfully finish our thesis work on schedule.

Dhaka
November 21, 2023

Grownthona Rahman

Sean Rahman

Rifat Bin Karim

Dewan Mubinul Haque

ABSTRACT

Cyberbullying is a type of online bullying that especially occurs on social media platforms. Similar to how English is used on social media, Bangla seventh most popular language in the world is also used there. Cyberbullying is on the rise as a result of how easily accessible the internet is. People occasionally fail to consider how their words, texts, or comments may affect the lives of others. Numerous studies on cyberbullying in English and other languages have been conducted to detect it. However, not much research has been done in Bangla on cyberbullying. In our thesis, we have worked on a dataset of 44001 Bangla comments collected from social media sites like Facebook, consisting of five distinct categories, which have been used to train and evaluate our model. Furthermore, an English dataset consisting of 24,783 tweets has been used. For text classification in our thesis, we have employed a variety of deep learning models, including LSTM (Long Short Term Memory), Bidirectional LSTM, a few hybrid models, and various transformer models. Also, for machine learning, we have employed models like Support Vector Machine (SVM), Multinomial Naïve Byes (MNB), Logistic Regression (LR), and Random Forest (RF). English and Bangla datasets both were used in machine learning. For the Bangla dataset, Random Forest (RF) performs remarkably well in both binary and multi-class classification. In binary, it is about 81% while in multi-class classification, it is 68%. For the English dataset, SVM performed better among all the models. We have only utilized the Bangla dataset for deep learning in addition to machine learning. A translation model, banglat5_nmt_en_bn was used to translate the English texts into Bangla. After translating, we combined the translated texts with the main dataset and fit them into XLM-ROBERTA Large, IndicBERT, and mBERT. After measuring the results for both balanced data and unbalanced data, we found that IndicBERT has the best performance with an accuracy of 93%. For multi-class classification, we have worked only with the original Bangla dataset and fit it into XLM-ROBERTA Base, DistilBERT, BanglaBERT, LSTM, Bi-LSTM, CNN+LSTM, and CNN+Bi-LSTM. For XLM-ROBERTA, both balanced and unbalanced data performed well having the highest accuracy of 92% which is rather prominent among various hybrid models and transformers models.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Introduction

In today's world, social media is a vital component in our day-to-day life. It is a modern and simple method for people to publicly share their feelings and opinions and interact online. In our technologically advanced society, cyberbullying has emerged as a disturbing phenomenon. Online bullying involves using digital communication platforms such as Facebook, Instagram, Snapchat, Tik Tok, online forums, and online gaming communities to torment, intimidate, or humiliate others. It is a type of aggressive conduct that uses the internet's anonymity and accessibility to target victims. Sending threatening or insulting messages, disseminating rumors, sharing humiliating photographs or videos without permission, impersonating an individual online, and excluding individuals from online communities are all examples of cyberbullying. [1] According to the First Site Guide, more adolescents were bullied on Instagram than on any other platform, at 42 percent. Facebook is in second place with 37%, followed by Snapchat, WhatsApp, and YouTube with 31%, 12%, and 10%, respectively. [2] Victims of cyberbullying frequently experience emotional distress, low self-esteem, anxiety, depression, and even suicidal ideation in extreme instances. According to a Pew Research Center poll performed from April 14 to May 4, 2022, 46% of American teenagers aged 13 to 17 claim to have experienced at least one of the six cyberbullying activities. [3]

Bengali has 265 million native speakers and is the 7th most widely spoken language in the world. [4] As of April 2023, there were 59004100 Facebook users in Bangladesh, or 33.6% of the country's total population. Among them, the majority was held by 66.1% men, and the remaining 33.9% were women. The largest user group, which is 26400000, were those between the ages of 18 and 24. [5] According to the Bangladesh Telecommunication Regulatory Commission (BTRC), an astounding 80 percent of Bangladesh's internet consumers

are on Facebook. [6] Due to the growing use of the internet and social media, like many other nations, Bangladesh is not immune to the expanding problem of cyberbullying. Due to the increasing availability of smartphones and the internet, social media platforms are now an integral part of the lives of Bangladeshis, particularly the younger generation. In addition to providing a platform for communication and expression, social media also allows for the possibility of misuse and abuse. Based on a study titled "Cyberbullying Against Girls and Women on Social Media" by the Bangladesh Institute of ICT in Development (BIID), Bangladesh has 70 million internet users and 22 million active social media users, a significant portion of whom are girls and women. [7] This study reveals that 33% of girls in rural areas and 64% of girls in urban areas receive sexually explicit videos, texts, and images as cyberbullying victims, with an overall victimization rate of 80%. [8]

1.2 Motivation

Cyberbullying is a type of emotional bullying that occurs online and on social media platforms. Bengali, the seventh most popular language in the world, is used on social media in a similar way to English. As the internet is so accessible, there is an increase in cyberbullying. People are freely expressing their opinions. They occasionally fail to consider whether their words, texts, or comments will have a negative effect on the lives of others. As a result, victims frequently feel depressed, self-harm, and occasionally even attempt suicide. We can now use machine learning, and deep learning approaches to detect cyberbullying on social media platforms like Facebook, Snapchat, and Twitter with the help of natural language processing. Numerous studies on cyberbullying in English have been conducted to stop hate speech using these approaches. However, there is not much research being done in Bangla on cyberbullying. Cyberbullying cases can be identified by selecting a suitable text classification model.

Chapter 2

Literature Review

2.1 Review of related papers

In the paper titled "Bangla hate speech detection on Social Media using attention-based Recurrent Neural Network" [9] proposed an encoder–decoder-based machine learning model for the classification of users' Bengali comments on Facebook pages. They used a dataset of 7,425 Bengali comments that were divided into seven different categories, including hate speech, aggressive comments, religious hatred, ethnic attacks, religious comments, political comments, and suicidal comments. The comments were gathered from some popular public pages in Bangladesh using the Facebook Graph API. Here, 1D convolutional layers were applied to the comments in order to extract and encode local characteristics. For data preprocessing, a variety of techniques were used. The Bangla Emot Module was utilized to extract data from emoticons, and this assisted in identifying the particular category of hate speech. Then, sentences in Bangla were broken up into words using the tokenization process. TF-IDF vectorization and word embedding were both utilized to extract features. The dataset was split into a training set (80%) and a testing set (20%). Lastly, for predicting hate speech Supervised machine learning algorithms like LSTM, and GRU-based decoders were used. An accuracy of 74% was obtained from the mentioned algorithms. After that, the model was enhanced with 77% accuracy by the attention mechanism.

The authors of "Cyberbullying Detection Using Deep Neural Network from Social Media Comments in Bangla Language" [10] proposed a binary and multiclass classification model using a hybrid neural network for the detection of bully expression in the Bengali language. Their proposed model can classify the non-bully sentence as well as different categories of bullies successfully. They used 44,001 users' comments from well-known Facebook pages, which were divided into the following five categories: non-bully, sexual, threat, troll, and religious. For them, the binary classification model gives 87.91% accuracy. After collecting

the dataset, they preprocessed it. Then, through the process of word embedding, they fed them into a hybrid neural network model with various variations. Finally, they evaluated the performance. They built a multiclass classification model to identify all the categories of harassment and got 79.29% accuracy. They also used an ensemble technique to increase this accuracy through binary classification and got 85% accuracy for classifying into different categories of harassment. Their dataset was up to date and consisted of a decent amount of data, which enabled them to get high-precision results. However, their 'threat' category had comparatively lower accuracy than other categories in multiclass classification as a result of its low recall value. But the precision was very high in this category. So, it doesn't incorrectly label any other comments as threats. It also produced false positive results for relatively longer and more complicated sentences.

In the next paper "Cyberbullying Detection using Machine Learning from Social Media Comments in Bangla Language" [11] investigated the performance and accuracy of numerous frequently used machine learning approaches on Bangla cyberbullying comments that were collected from Facebook. In the case of the Random Forest classifier, they got 96% accuracy. At first, they pre-processed the dataset that was collected. Then, feature extraction was done. They have used word embedding through Word2Vec as a feature selection to send the pre-trained small dataset from the large repository to the Random Forest classifier for classification. Their approach gives an F1-score of 76%, precision of 81%, and recall of 74%. And it correctly predicts 85% of non-bullying comments. As well as 80% sexual, 48% threat, 73% troll remarks, and 82% religious.

For the paper "Social media bullying detection using machine learning on Bangla Text" [12] the authors carried out the training and testing process in two phases. In the first phase, only text-based features (posts/comments) were considered. It achieved an accuracy of 95.40% for the support vector machine-based algorithm on Bangla text after doing 10-fold cross-validation by separating bullied text contents from the non-bullied ones. In the second phase, both text-based features and user information (location, age, and gender) were considered in training the classification models, which improved the performance of the algorithm. In this case, KNN (3-Nearest) achieved the highest accuracy of 97.73%, and SVM came in second with 97.27%. SVM, therefore, appears to be effective for both approaches. Their experiment included 2,400 Bangla texts collected from social media posts, of which 10% were bullies

The paper "Abusive Comments Detection in Bangla-English Code-mixed and Transliterated Text" [13] proposed a method to detect abusive comments written in Bangla text or code-mixed Bangla-English text or transliterated Bangla text. In order to create the classification

model, they used three classification algorithms—Support Vector Machine, ensemble models—Random Forest, and Adaboost. Their dataset consisted of comments along with the number of likes collected from public Facebook pages. For training and testing, they randomly divided the total dataset into 80-20 groups. After that, 10-fold cross-validation was used to verify the training dataset. In the preprocessing step, with the usual processes that are carried out to clean the dataset from noise, they detected profanity words and replaced them with the words that they were masking. Then, to create their classifier model they extracted the following features-N-grams, profane words, number of profane words, emojis, number of emojis in 24 predefined categories, number of likes, and sentiment scores. They experimented with multiple kernels for the SVM classifier, including the linear kernel, radial basis function (RBF) kernel, and polynomial kernel. The maximum features and the number of trees were changed for the Random Forest classifier in order to discover the best splitting. They changed the learning rate and the number of estimators for the Adaboost classifier. From their result metrics we can see that, Random Forest offered the best accuracy and the worst recall. On the other hand, Adaboost had the best recall but it offered the worst accuracy and precision which means it correctly detected the most abusive comments but it incorrectly classified some non-abusive comments as abusive. Random Forest could identify the least number of abusive comments but it correctly labeled the most amount of non-abusive comments.

The authors of "An Application of Machine Learning to Detect Abusive Bengali Text" [14] studied machine learning algorithms like Multinomial Naïve Bayes (MNB), Random Forest (RF), Support Vector Machine (SVM) with Linear, Radial Basis Function (RBF), Polynomial and Sigmoid kernel and compared with unigram, bigram and trigram based CountVectorizer and TfidfVectorizer features. The results show that the SVM Linear kernel with trigram TfidfVectorizer features perform the best when compared with CountVectorizer features. Their abusive text Bengali dataset was collected from Facebook comments. Each Bengali Unicode comment was then manually classified as abusive or not abusive. 500, 1000, 1500, 2000, and 2500 comments from the dataset were used for testing. To train the ML algorithms unigram, bigram, and trigram features were extracted from all the comments and vectorized using CountVectorizer and TfidfVectorizer. The results were then validated using a 10 times 10-fold cross-validation method. After using CountVectorizer, in the case of unigrams, SVM with Linear kernel performs the best in terms of accuracy, whereas SVM with Sigmoid kernel performs the worst. The performance of RF, MNB, and SVM (Linear) increases when the number of training data goes higher. In the case of bigram, MNB shows the best accuracy and SVM with Linear kernel are very close to it. All algorithms improve in performance as the number of datasets increases, with the exception of the SVM Polynomial kernel, which remains almost the same. In the case of a bigram, MNB has the highest accuracy, but SVM with a Linear kernel is fairly close. The performance of all methods increases with the num-

ber of datasets except for the SVM Polynomial kernel. In the case of a trigram, RF, MNB, and SVM with Linear Kernel perform significantly better than SVM with RBF, Polynomial, and Sigmoid. MNB and RF have the highest accuracy of any experiment performed in this study. The performance of SVM RBF and Sigmoid kernels decreases as the number of datasets increases, but the Polynomial kernel remains constant.

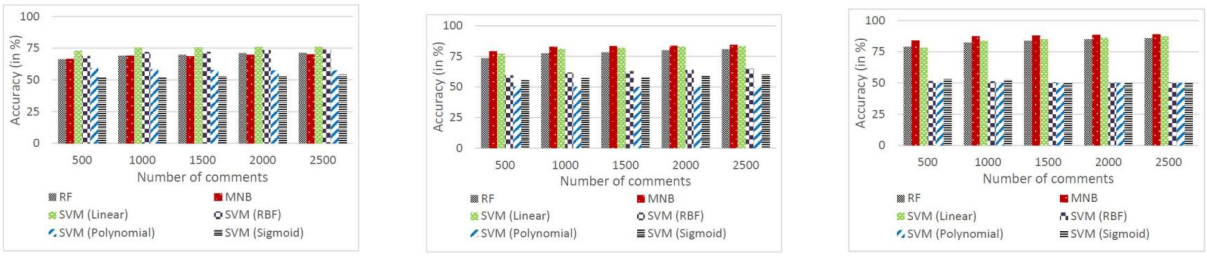

Figure 2.1: Unigram, Bigram, Trigram with CountVectorizer perspective

After using TFIDFVectorizer in the case of unigram, SVM with Linear kernel has the highest performance. On the other hand, MNB's results are poor. Also, SVM with the RBF and Sigmoid kernel has poor accuracy. In case of Bigram, SVM with Linear Kernel shows the best accuracy. The final three positions are taken by SVM with RBF, Polynomial, and Sigmoid. In the case of Trigram, the SVM Linear kernel classifier produced the best result out of all the experiments. In terms of accuracy, MNB and RF come in second and third, respectively.

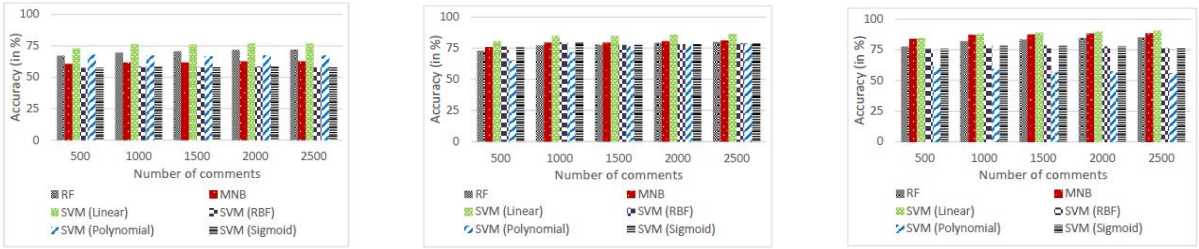

Figure 2.2: Unigram, Bigram, Trigram with TD-IDF perspective

According to the results of the studies, SVM with a linear kernel consistently outperforms other methods. RBF, Polynomial, and Sigmoid kernels of SVM, on the other hand, cannot perform quite well. Also, MNB has very good accuracies in most cases and RF provides some accurate results.

"Deployment of Machine Learning and Deep Learning Algorithms in Detecting Cyberbullying in Bangla and Romanized Bangla Text: A Comparative Study" [15] prepared three datasets from social media (YouTube) for Bangla, Romanized Bangla, and a combination of both. The three datasets contained 5000 Bangla, 7000 Romanized Bangla, and a combination of 12000 Bangla and Romanized Bangla texts. They manually labeled their datasets

as bullying and not bullying. In their study, they used four Machine Learning classification models, such as Multinomial Naïve Bayes, SVM, Logistic Regression, and XGBoost. For Deep Learning they used CNN, LSTM, BLSTM, and GRU. The CNN algorithm achieved 84% accuracy in the case of the Bangla dataset, giving the best performance. For the other two datasets, Multinomial Naïve Bayes performed best by achieving 84% accuracy in the Romanized Bangla dataset and 80% accuracy in the combined dataset. In preprocessing, with other processes of removing noise from the dataset, they removed emojis too. However they did not remove any stop words or perform stemming and lemmatization as the comments consisted mostly of many local languages which helped them to get maximum features available. For the feature extraction, they used TF-IDF which was used in the case of Machine Learning Algorithms. Word Embedding was also used by them and by using one-hot representation, the vectors were used as input for the embedding layer of all Deep Learning algorithms. Following feature selection and preprocessing, they split all three datasets into 80% and 20% for training and testing respectively.

In the paper titled "Using machine learning to detect cyberbullying" Reynolds et al. [16] found that both the C4.5 decision tree and 3-nearest neighbor classifiers can reach a recall of 78.5% on the text-based dataset collected from Formspring.me, a question-and-answer-based social network. So, He made an experiment containing 2696 posts where 196 received a final class label of “yes,” indicating the presence of cyberbullying (7.2% of the posts). At first, they preprocessed the dataset. Then they tested the model created with the 8-weight NORM training set and J48 algorithm, and they obtained a true positive accuracy of 61.6% and an overall accuracy of 81.7%

The authors of "Cyberbullying detection: a step toward a safer Internet yard" [17] used a cross-system analysis of the users’ behavior monitoring their reactions in different online environments. They used a dataset consisting of more than 381,000 posts in about 16,000 threads. Overall, 34% of posts are written by female and 64% by male authors, and analyzed the use of foul words in 100,000 randomly selected posts from the dataset. At First the corpus with posts written by both male and female users as our dataset. In the next step, they trained the classifier separately for each gender group. Then calculated the final result, based on the proportion of each group in the whole corpus (34% female, and 66% male). At last, they used 10-fold cross-validation and calculated corresponding precision, recall, and F-measure. The final result after the Experiment is shown in Table 2.1

Table 2.1: Result table [17]

Feature Used in Classifier	Precision	Recall	F-Measure
Baseline	0.31	0.15	0.20
Gender-Specific	0.43	0.16	.23
Female Specific	0.40	0.05	0.08
Male Specific	0.44	0.21	0.28

In the paper titled "Learning from bullying traces in social media" [18] the authors presented evidence that social media, with appropriate natural language processing techniques, can be a valuable and abundant data source for the study of bullying in both worlds. They used different models to get the highest accuracy on the dataset. The combination of SVM(linear) + 1g2g achieves an average accuracy of 79.7%. SVM(linear) + 1g2g achieves 81.3%, which is significantly better (t-test, $p = 4 \times 106$). It shows that including bigrams can significantly improve the classification performance. SVM(linear) + 1g2g POS achieves 81.6%, though the improvement is not statistically significant ($p = 0.088$), which indicates that POS coloring does not help too much on this task. SVM(RBF) gives a similar performance, Logistic Regression is slightly worse and Naive Bayes is much worse, for a large range of training set sizes. In summary, SVM(linear) + 1g2g is the preferred model because of its accuracy and simplicity. We also note that these accuracies are much better than the majority class baseline of 61%. On the holdout set, SVM(linear) + 1g2g achieves precision P=0.76, recall R=0.79, and F-measure 0.77. Linear CRF achieved an accuracy of 87%. The best cross-validation accuracy of 89% is obtained by SVM(linear) + 1g2g

The authors of “Modeling the Detection of Textual Cyberbullying,” [19] decomposed the overall detection problem into the detection of sensitive topics, lending itself to text classification sub-problems. They experiment with a corpus of 4500 YouTube comments, applying a range of binary and multiclass classifiers. They find that binary classifiers for individual labels outperform multiclass classifiers. Their dataset was subjected to three operations: removal of stop-words, stemming, and removal of unimportant sequences of characters. They divided datasets into three clusters which are 50% training, 30% validation, and 20% test data. Each dataset was preprocessed to clean and massage the data. Next, they applied different types of models to get the accuracy. Using Naive Bayes they get an overall 63% accuracy. Another method called Rule-based Jrip was used and got the same accuracy of 63%. Another method called Tree-based J48 gets a minimum accuracy of 61%. And the overall highest accuracy they got was the SVM model which is 66.7%.

The next paper is titled as “A Comparative Analysis of Machine Learning Techniques for Cyberbullying Detection on Twitter" [20] Here, the authors made an effort to detect cyberbullying by gathering a global dataset of 37,373 distinct tweets from Twitter. They used seven machine learning classifiers which are Logistic Regression (LR), Light Gradient Boosting Machine (LGBM), Stochastic Gradient Descent (SGD), Random Forest (RF), AdaBoost (ADB), Naive Bayes (NB), and Support Vector Machine (SVM). Each of these algorithms was evaluated using accuracy, precision, recall, and F1 score as the performance metrics to determine the classifiers' recognition rates applied to the global dataset. The dataset has been divided into two parts. The first part contains 70% of the tweets used for training purposes and another 30% used for prediction purposes. They preprocess the dataset by cleaning texts, as well as spam content removal. Then they used TF-IDF and Word2Vec techniques for feature extraction. After that seven machine learning algorithms were constructed namely, LR, Light LGBM, SGD, RF, AdaBoost, Naive Bayes, and SVM. The SVM has achieved the lowest accuracy and precision of 74.7% and 75.4%. Another method called Multinomial NB has achieved low accuracy and precision with a detection rate of 81.39% and 0.7952, respectively, and they notice that the excellent recall levels out the low precision, giving a good F-measure score of 0.8754.RF and AdaBoost have achieved almost the same accuracy of 87.7%, but in terms of the F1 Score, RF performs better than AdaBoost.Finally, FR, SGD, and LGBM give the best accuracy. Among them, LR achieved the best accuracy and F1 score. Where the classification accuracy and F1 score are 90.57% and 0.9280.SGD achieved an accuracy of 90.6%, but the F1 score was lower than LR. However, the LGBM classifier achieved an accuracy of 90.55%, and the F1 score was 0.9271. So This means LR performs better than other classifiers in this investigation.

For the next paper “DEA-RNN: A Hybrid Deep Learning Approach for Cyberbullying Detection in Twitter Social Media Platform” [21] the authors presented a hybrid deep learning model, called DEA-RNN to detect Cyberbullying on Twitter social media networks. The DEA-RNN model is a combination of Elman type Recurrent Neural Networks (RNN) with an optimized Dolphin Echolocation Algorithm (DEA) for fine-tuning the Elman RNN's parameters and reducing training time. They used a dataset containing of 10000 tweets. They also implemented Bi-directional long short-term memory (Bi-LSTM), RNN, SVM, Multinomial Naive Bayes (MNB), and Random Forests (RF) to compare with the DEA-RNN model. The initial dataset includes 435764 with racism, insult, swear, and sexism-based keywords contributing about 130000 tweets but the dataset has some outliers. They need only English-language tweets so they remove the irrelevant tweets. After removing the irrelevant tweets about 10000 tweets are randomly selected containing 32 different cyberbullying keywords. Around 6,508 (0.65%) are non-cyberbullying, and 3492 (0.35%) are cyberbullying tweets. The selected tweets were labeled manually into two labels, either “0” non-cyberbullying or “1” cyberbullying. Then they preprocess the dataset. After that, for feature extraction,

they use NLP tools such as Word2Vec and TF-IDF, with the nouns, pronouns, and adjectives are considered as primary feature contents, whereas the adverbs and verbs provide additional information.Bi-LSTM, RNN, SVM, MNB, and RF have got average accuracy of 88.74%, 87.15%, 85.21%, 82.26%, and 83.45% respectively. Where the DEA-RNN model has obtained the highest average accuracy of 90.45%. Bi-LSTM, RNN, SVM, MNB, and RF have obtained the average precision of 87.9%, 86.62%, 84.25%, 80.01%, and 83.87% respectively. Where DEA-RNN has got 89.52%. For average recall Bi-LSTM, RNN, SVM, MNB, and RF, have obtained 87.52%, 85.9%, 82.72%, 78.89%, and 82.49%, respectively. But DEA-RNN scored 88.98%. The average F-MEASURE of Bi-LSTM, RNN, SVM, MNB, and RF have obtained 87.71%, 86.26%, 83.48%, 79.45%, and 83.17% and DEA-RNN has got 89.25% average F-measure. In terms of training time DEA-RNN model has less training time compared to other deep learning Bi-LSTM, RNN.

In this paper “Deep Learning Algorithm for Cyberbullying Detection,” [22] the authors proposed a novel algorithm CNN-CB that eliminates the need for feature engineering and produces better prediction than traditional cyberbullying detection approaches. CNN-CB is based on convolutional neural networks (CNN) and incorporates semantics through the use of word embedding. For the dataset, they use a total of 39,000 tweets retrieved from Twitter’s public timeline. For training purposes, 9000 bullying and 21000 nonbullying data are used and for testing purposes, 2700 bullying and 6300 non-bullying data are used. They use two different models for this experiment those are SVM and CNN-CB. For the SVM model, they get an accuracy of 81.32% and precision of 73% and for recall they get 70%.CNN-CB gives better accuracy of 93% and precision of 79% and for recall they get 81%. For this experiment, the deep learning model works way better than the normal machine learning model.

A paper called “An Application to Detect Cyberbullying Using Machine Learning and Deep Learning Techniques" [23] by Mitushi Raj Samridhi Singh Kanishka Solanki Ramani Selvanambi has also worked on the detection of cyberbullying. In this paper, their main motivation is to reduce cyberbullying at the time of covid-19 when people started using social media more often as a result cyberbullying increased rapidly. They have used a dataset that has three levels including English, Hindi, and Hinglish which is a Romanized version of Hindi and English text from different social media posts. After that, they have pre-processed the dataset by using different methods. Then the feature extraction was done. For the word embedding part, they have used two different methods which are GloVe and FastText.After that several deep learning approaches were used like LSTM, CNN+BIGRU, CNN+BILSTM, and BILSTM+BIGRU.Where for the LSTM model 87% is the highest accuracy using Sigmoid Function and Adam optimizer.CNN+BILSTM gives the highest accuracy of 91.35% before Hypertuning and 95.12% after Hypertuning. The other two models CNN+BIGRU and BIL-

STM+BIGRU give an accuracy of 93.69% and 88.53% respectively after Hypertuning.

In a paper called “BD-SHS: A Benchmark Dataset for Learning to Detect Online Bangla Hate Speech in Different Social Contexts ” [24] collected data from online social media and online streaming sites. They polled 20 undergraduate students from Shahjalal University of Science and Technology’s 17 different majors 12 men and 8 women in a brief survey. Every participant was between the ages of 20 and 24 are regular social media user. After completion of all surveys, they left with a dataset containing 50381 comments. Then the preprocess was done. For the feature extraction part, they use TF-IDF. Then for the word embedding part, they used Multilingual fasttext (MFT) which is a pretrained article in 157 languages, while BengFastText (BFT) is pretrained 250 million Bangla articles. It is important to note that Wikipedia articles are used to prettrain both of these models. Then they made some experiments by implementing some machine learning and deep learning models like SVM + C, Bi-LSTM+BFT, Bi-LSTM+MFT, and Bi-LSTM+IFT.The highest accuracy is obtained by Bi-LSTM+IFT which is 85.08%. Then the second highest is obtained by SVM + C which is 84.27%. The third position is Bi-LSTM+MFT which is 82.17%. The last position is for Bi-LSTM+BFT which is 75.21%. Overall Bi-LSTM+IFT performs the best

“A Multilingual System for Cyberbullying Detection: Arabic Content Detection using Machine Learning is a multilingual system” [25] to detect something abnormal from social media based on the Arabic language. They have worked with two different dataset one is ArabicTweets which contains 35373 unique data after removing all duplicates another dataset, the English dataset contains 91431 data. All the data have been collected from different tweets. They use binary classification for the classification part. After completing all the preprocessing by removing stopwords, duplicates, numerical numbers, etc the feature extraction is done by TF-IDF. They use translation to translate the English language to Arabic. After that, they try to classify them by using binary classification. For the SVM model the precision is 93.4% which is the most and for Naive Bayes the precision is 90.1%.

In the paper “Cyberbullying Detection: Hybrid Models Based on Machine Learning and Natural Language Processing Techniques" [26] collected the dataset from Wikipedia which contains 159,689 data where 15365 is toxic and 144,324 are non-toxic data. For the machine learning models first, they preprocess the data then they move onto the feature extraction part. For machine learning, they employ four techniques for classical machine learning models: TF-IDF character bigram and trigram, TF-IDF word bigram and trigram, and count vectorization. For the shallow neural network, they use GloVe, FastText, and Paragram as the embedding representations. For the classification part, they both use traditional machine learning and deep learning models like SVM, XGBoost, Naïve Bayes, CNN, LSTM, Bi-LSTM,

Att-BiLSTM, BERT, etc. Traditional machine learning models like the SVM model get 95.1% accuracy where XGBoost gets 94.7% accuracy Naive Bayes gets an accuracy of 94.8% and lastly Logistic regression gets 94.9% accuracy. For the neural network model, Bi-GRU with GloVe embeddings is the best-performing model with an accuracy of 96.98% and 98.56% F1-score. On the other hand, Bi-LSTM models have proven to be very effective. The highest F1-score is 98.69% with GloVe embeddings, which is followed by 98.65 with CNN using Paragram and Attention-BiLSTM.

2.2 Comparison of related papers

A comparison table that includes the essential details from every paper that we have looked at so far is given below:

Table 2.2: Summary of related papers.

Reference Work	Training Data	Classification Algorithm	Performance
Using machine learning to detect cyberbullying [16]	Contains 2696 posts	Algorithms: J48	Maximum Accuracy of 81.7% (J48)
Learning from bullying traces in social media [18]	1762 tweets sampled uniformly from the enriched dataset	Algorithms: Naive Bayes,Logistic Regression, SVM (Linear), SVM (RBF)	Maximum Accuracy 89% is obtained by SVM(linear) + 1g2g
Modeling the Detection of Textual Cyberbullying [19]	Contains 4500 YouTube comments	Algorithms: Naive Bayes, JRIP, J48, SVM	Maximum Accuracy of 66.7% (SVM)
Deep Learning Algorithm for Cyberbullying Detection [22]	Dataset contains 39,000 tweets	Algorithms: SVM, CNN-CB	Maximum Accuracy of 93% (CNN-CB)
A Comparative Analysis of Machine Learning Techniques for Cyberbullying Detection on Twitter [20]	Dataset contains 37,373 distinct tweets from Twitter	Algorithms: LR, Light Gradient Boosting Machine, Stochastic Gradient Descent, Random Forest, AdaBoost(ADB), Naive Bayes(NB), and SVM	Maximum Accuracy of 90.57% (LR)
Cyberbullying Detection Using Deep Neural Network from Social Media Comments in Bangla Language [10]	Dataset contains 44,001 users' comments from well-known Facebook pages	Algorithms: Hybrid Neural Network and Ensemble Technique	Maximum Accuracy of 87.91% in binary and in multiclass accuracy of 79.29%

Continuation of table 2.2.

Reference Work	Training Data	Classification Algorithm	Performance
DEA-RNN: A Hybrid Deep Learning Approach for Cyberbullying Detection in Twitter Social Media Platform [21]	Dataset contains 10000 tweets are randomly selected where 6,508 (0.65%) are non-cyberbullying, and 3492 (0.35%) are cyberbullying	Algorithms: Bi-LSTM, RNN, SVM, MNB, RF and DEA-RNN	Maximum Accuracy of 90.45% (DEA-RNN)
Social media bullying detection using machine learning on Bangla text [12]	Dataset consists of 2,400 Bangla texts collected from social media posts	Algorithms: KNN, SVM	Maximum Accuracy of 97.73% (KNN)
An application of Machine Learning to Detect Abusive Bengali Text [14]	Dataset contains abusive or not abusive. 500, 1000, 1500, 2000 and 2500 comments	Algorithms: SVM(linear), SVM (RBF), SVM(Polynomial), SVM(Sigmoid)	Maximum Accuracy of 77.3% (SVM(linear))
Deployment of Machine Learning and Deep Learning Algorithms in Detecting Cyberbullying in Bangla and Romanized Bangla Text: A Comparative Study [15]	5000 Bangla, 7000 Romanized Bangla and a combination of 12000 Bangla and Romanized Bangla texts	Algorithms: Multinomial NB, SVM, XGBoost, Logistic Regression	Accuracy of 84% in Bangla dataset (CNN), 84% and 80% in Romanized Bangla and combined dataset. (Multinomial Naïve Bayes)
Bangla hate speech detection on social media using attention-based recurrent neural network [9]	Dataset consists of 7,425 Bengali comments	Algorithms: LSTM, and GRU-based decoders	Maximum Accuracy of 77% (Attention Mechanism)
Cyberbullying Detection using Machine Learning from Social Media Comments in Bangla Language [11]	Dataset contains 44,001 users' comments from well-known Facebook pages	Algorithm: Random Forest	Accuracy of 96% (Random Forest)
Abusive Comments Detection in Bangla-English Code-mixed and Transliterated Text [13]	Dataset contains more than 381,000 posts	Algorithm: SVM, Random Forest, and Adaboost	Best accuracy and worst recall (Random Forest)

Continuation of table 2.2.

Reference Work	Training Data	Classification Algorithm	Performance
An Application to Detect Cyberbullying Using Machine Learning and Deep Learning Techniques [23]	English, Hindi and Hinglish dataset collected from different social media posts	Algorithms: LSTM, CNN+BIGRU, CNN+BILSTM, BILSTM+BIGRU	Maximum accuracy of 91.35% before Hypertuning and 95.12% after Hypertuning (CNN+BILSTM)
BD-SHS: A Benchmark Dataset for Learning to Detect Online Bangla Hate Speech in Different Social Contexts [24]	Dataset contains 50381 comments	Algorithms: SVM+ C, Bi-LSTM+BFT, Bi-LSTM+MFT and Bi-LSTM+IFT	Accuracy of 85.08% (Bi-LSTM+IFT)
A Multilingual System for Cyberbullying Detection: Arabic Content Detection using Machine Learning is a multilingual system [25]	The dataset consists of ArabicTweets which contains 35373 data another dataset which is English dataset contains 91431 data	Algorithms: SVM, Naive Bayes	Maximum Accuracy of 93.4% (SVM)
Cyberbullying Detection: Hybrid Models Based on Machine Learning and Natural Language Processing Techniques [26]	Dataset contains 15365 toxic and 144,324 non-toxic data	Algorithm: SVM, XGBoost and Naïve Bayes, CNN, LSTM, Bi-LSTM, Att-BiLSTM, BERT	Accuracy of 96.98% (Bi-GRU with GloVe embeddings)

Chapter 3

Background Study

This section describes the methodology of the Cyber Bullying Detection and gathers information about the subject in order to gain a deeper understanding of it.

3.1 Machine learning algorithm for text classification

Text classification with machine learning is usually much more accurate than human-crafted rule systems, especially on complex NLP classification tasks. Several machine learning algorithms are frequently used for text classification. Additionally, machine learning classifiers are simpler to maintain, and one can always tag new examples to pick up new skills. Naive Bayes is a probabilistic classifier that works well with small to medium-sized datasets because of its efficiency and ease of use. Support Vector Machines construct hyperplanes in high-dimensional spaces, offering effectiveness and versatility. A linear model that can be easily understood and applied to data that can be divided linearly is logistic regression. By combining multiple decision trees, the ensemble method Random Forest is robust and minimizes overfitting. Gradient Boosting Models like XGBoost and LightGBM provide high predictive accuracy. Experimenting with various algorithms is common to achieve optimal performance in text classification tasks. The choice of algorithm depends on factors like dataset size, data complexity, and task requirements.

3.1.1 Support Vector Machine - SVM

Support Vector Machine is proven to be a supervised machine learning method. This is considered to be used in solving both regression and classification problems. It is typically employed as a classifier.An additional name for the Support vector machine algorithm is a max-margin classifier. It is an effective tool for machine learning and has been extensively

used in many applications like text classification, hand-written digit recognition, and facial expression recognition. Over other machine learning techniques, the support vector machine has various benefits, including resistance to noise and the capacity to handle large datasets. In a high- or infinite-dimensional space, a support vector machine creates a hyperplane or set of hyperplanes that can be used for classification, regression, or other tasks like outlier detection. [27] Generally speaking, the higher the margin, the smaller the classifier's generalization error, so a decent separation is obtained by the hyperplane that has the largest distance to the nearest training data point of any class (the so-called functional margin). [27]

SVM is effective in high dimensional cases, and its memory is efficient as it uses a subset of training points in the decision function called support vectors. In SVM, different kernel functions can be specified for the decision functions, and it is possible to specify custom kernels.

There are two types of kernels for SVM. The kernels are **Linear** and **Nonlinear**. The **Linear** SVM provides information about hard margin and soft margin, which is helpful in classifying the data.The **Non-Linear** kernels are not linear. Some of the Non-Linear kernels are :

- Polynomial(homogeneous).
- Polynomial (inhomogeneous)
- Gaussian radial basis function
- Hyperbolic tangent

3.1.2 Multinomial Naive Bayes

A probabilistic method for building data classification models is called naive Bayes. It deals with probability as the "likelihood" that data belongs to a certain class and is often used as an alternative to decision tree forests and distance-based K-means clustering. There are two types of naive Bayes' Gaussian and Multinomial models.

The Multinomial Naive Bayes algorithm is a probabilistic learning method that is mostly used in Natural Language Processing (NLP). The method, which guesses the tag of a text such as an email or newspaper article, is based on the Bayes theorem. For a given sample, it determines the probabilities of each tag and then outputs the tag with the highest probability. An effective algorithm for issues involving several classes and text data analysis is multinomial naive bayes. It's critical to first understand the Bayes theorem concept in order to understand how the Multinomial Naive Bayes theorem functions. The Bayes theorem,

developed by Thomas Bayes, determines the likelihood that an event will occur based on previously known conditions. Its foundation is the following equation:

$$P(A|B) = \frac{P(A) \cdot P(B|A)}{P(B)} \tag{3.1}$$

Here, P(B) = prior probability of B
P(A) = prior probability of class A
P(B|A) = occurrence of predictor B given class A probability

This formula helps in calculating the probability of the tags in the text.

The Multinomial Naive Bayes which uses the Naive Bayes algorithm has the following advantages:

- It is easy to implement as we only have to calculate probability.
- We can use this algorithm on both continuous and discrete data.
- It is simple and can be used for predicting real-time applications.
- It is highly scalable and can easily handle large datasets.

3.1.3 Logistic Regression

Logistic regression is a popular statistical model used for binary classification tasks, where the goal is to predict the probability of an instance belonging to a certain class. It is a probabilistic classifier similar to the Naïve Bayes.It's called "logistic" regression because it uses the logistic function to model the relationship between the independent variables and the dependent variable.
Consider a single input observation x, which we will represent by a vector of features [x1, x2,...,xn] The classifier output y can be 1 (meaning the observation is a member of the class) or 0 (the observation is not a member of the class). We want to know the probability P(y=1|x) that this observation is a member of the class. So perhaps the decision is "positive sentiment" versus "negative sentiment", the features represent counts of words in a document, P(y=1|x) is the probability that the document has positive sentiment, and P(y = 0|x) is the probability that the document has negative sentiment.

3.1.4 Weight and Bias

Logistic regression solves this task by learning, from a training set, a vector of weights and a bias term. Each weight wi is a real number, and is associated with one of the input features

xi. The weight wi represents how important that input feature is to the classification decision, and can be positive (providing evidence that the instance being classified belongs in the positive class) or negative (providing evidence that the instance being classified belongs in the negative class). Thus we might expect in a sentiment task the word awesome to have a high positive weight, and bias term abysmal to have a very negative weight. The bias term, also called the intercept, is the intercept of another real number that's added to the weighted inputs.

To make a decision on a test instance— after we have learned the weights in training— the classifier first multiplies each x_i by its weight w_i , sums up the weighted features, and adds the bias term b. The resulting single number z expresses the weighted sum of the evidence for the class.

$$z = \left(\sum_{i=1}^{n} w_i x_i\right) + b \tag{3.2}$$

The above equation can be represented using the dot product notation from linear algebra. The dot product of two vectors a and b, written as a · b is the sum of the products of the corresponding elements of each vector. Thus the following is an equivalent formation of the above equation:

$$z = w \times x + b \tag{3.3}$$

Because this is simply the below vector calculation

$$\begin{aligned} &f(\mathbf{x}) = w_0 x_0 + w_1 x_1 + w_2 x_2 + \ldots w_d x_d = \mathbf{w}^T \mathbf{x} \\ &w_0, w_1, \ldots w_k - \text{ parameters (weights)} \end{aligned} \tag{3.4}$$

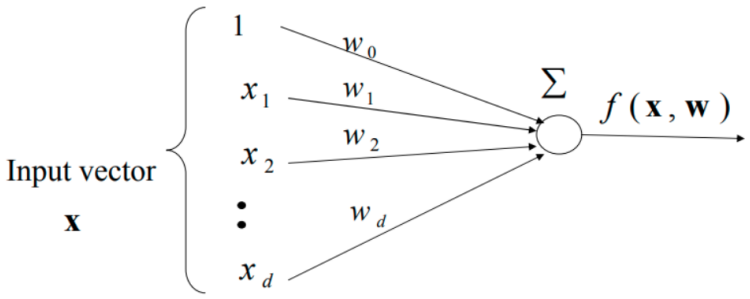

Figure 3.1: Weight and Bias

3.1.5 Logistic Function

The logistic function is also known as the sigmoid function. It has an S-shaped curve that asymptotically approaches these boundary values. The sigmoid has a number of advantages;

it takes a real-valued number and maps it into the range [0,1], which is just what we want for probability. Because it is nearly linear around 0 but flattens toward the ends, it tends to squash outlier values toward 0 or 1. To create a probability, we'll pass z through the sigmoid function, (z). The sigmoid has the following equation:

$$y = \sigma(z) = \frac{1}{1+e^{-z}} = \frac{1}{1+\exp(-z)} \tag{3.5}$$

If we apply the sigmoid to the sum of the weighted features, we get a number between 0 and 1. To make it a probability, we just need to make sure that the two cases, p(y = 1) and p(y = 0), sum to 1.

Now we have an algorithm given an instance x computes the probability P(y = 1|x). For a test instance x, we say yes if the probability P(y = 1|x) is more than .5, and no otherwise. We call .5 the decision boundary.

The S-shaped curve of the sigmoid function helps us to gain proper knowledge about how the sigmoid function works. The figure and the equation are given below:

$$\hat{y} = \begin{cases} \mathbf{1} \text{ if } \boldsymbol{P(y=1 \mid x) > 0.5} \\ \mathbf{0} \text{ otherwise} \end{cases} \tag{3.6}$$

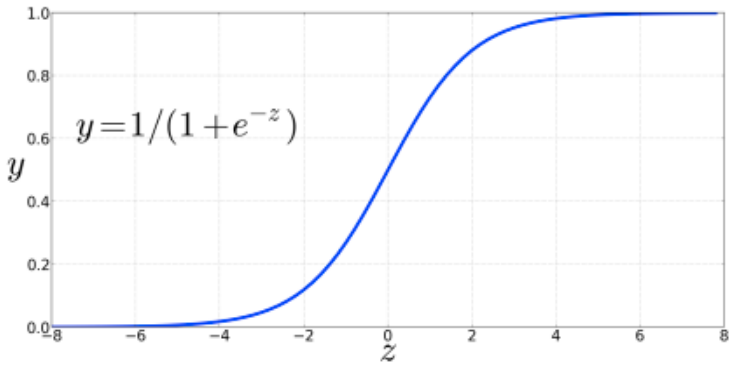

Figure 3.2: Sigmoid Curve

3.1.6 Decision Boundary

In logistic regression, a decision boundary is used to separate the two classes. It is a line that separates the instances predicted for one class from the instances predicted for the other class. The decision boundary is derived from the logistic function or activation function "Sigmoid".

3.1.7 Training Process

The training process involves optimizing the model parameters to minimize the difference between the predicted probabilities and the actual class labels in the training data. This optimization is typically done using an algorithm called gradient descent.

3.1.8 Classification

To obtain a binary prediction, we need to convert the predicted probabilities into discrete class labels. We can do this by choosing a threshold value (e.g., 0.5) and classifying instances with predicted probabilities above the threshold as one class and those below as the other class.

3.1.9 Random Forest

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique.It can be applied to ML problems involving both classification and regression. It is built on the idea of ensemble learning, which is a method of combining multiple classifiers in order to solve complex problems and enhance model performance. According to what its name indicates, "Random Forest" is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset." Instead of depending on a single decision tree, the random forest takes predictions from each tree and predicts the result based on the votes of the majority of predictions. The greater number of trees in the forest leads to higher accuracy and prevents the problem of over-fitting.

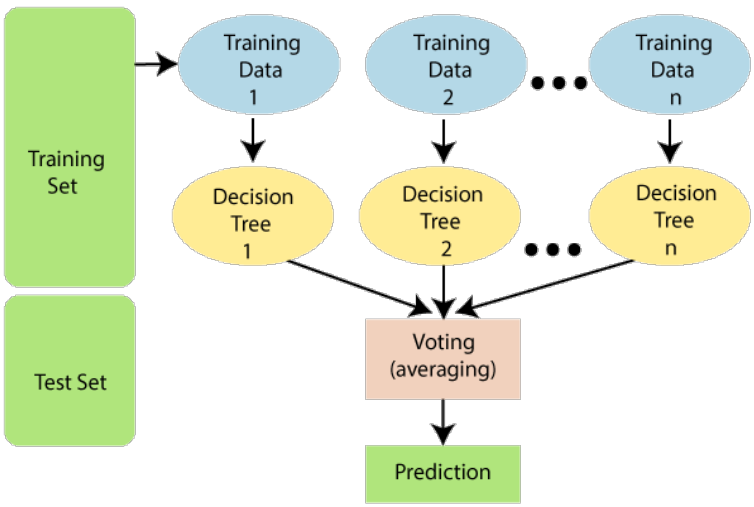

Figure 3.3: Visualization of a Random Forest Model Making a Prediction

Some decision trees may predict the correct output, while others may not, because the random forest combines numerous trees to forecast the class of the dataset. But when all the trees are combined, they forecast the right result. The random forest algorithm works by creating an ensemble of decision trees, where each tree is trained on a different subset of the training data. The steps in the procedure are as follows:

- Datasampling: A technique known as bootstrapping is used by random forest to choose at random a portion of the training data with replacement. This implies that some instances might be repeated whereas others may not appear in a particular subset. The training sets of each tree are different thanks to this sampling strategy.
- DataFeature sampling: At each node of the decision tree, a random selection of features is chosen to be considered rather than all features. By doing so, randomness is introduced and a few strong traits keep the decision-making process from being dominated.
- DataBuilding decision trees: Several decision trees are generated using the bootstrapped data and feature subset. Recursively splitting the data according to the chosen features creates each tree. The splitting is carried out in a way that maximizes a specific criterion, such as mean squared error for regression or Gini impurity or information gain for classification.
- DataVoting and prediction: After all the trees have been constructed, each tree makes a prediction on its own. In the case of classification, the final prediction is made for the class that receives the most votes across all of the trees. The anticipated values from all the trees are averaged for regression.

Example: [28] Let's use an example where the dataset is made up of several fruit photos. Therefore, the Random forest classifier receives this dataset. Each decision tree is given a portion of the overall dataset. Each decision tree generates a prediction result during the training phase and the Random Forest classifier predicts the outcome based on the majority of results when a new data point is encountered. Consider the figure below:

Random Forest offers several advantages. They are given below:

- Robustness: Compared to individual decision trees, Random Forest is less prone to overfitting and can tolerate noisy and missing data.
- Feature importance: It provides a measure of feature importance and identifies the features that are most crucial to the model's ability to forecast the future. Outlier detection: By comparing predicted and actual values, Random Forest is able to identify misfits.

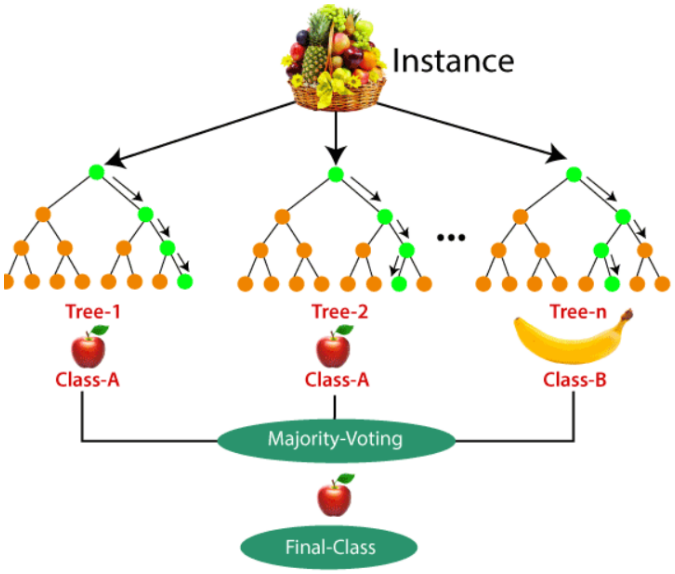

Figure 3.4: A visual representation of fruits classification using Random Forest.

- Scalability: It is capable of handling big datasets with plenty of features.
- Versatility: Both classification and regression issues can be solved with Random Forest.

3.2 Deep Learning Algorithms for text classification

Deep learning is a subset of machine learning that is based on artificial neural networks. They are designed to learn from large amounts of data and solve complex problems. A neural network with only one layer can still generate approximate predictions, but it can be optimized and refined for accuracy with the help of more hidden layers. They can be used for supervised, unsupervised, and reinforcement learning. For text classification in our thesis, we employed a variety of deep learning models, including Convolutional Neural Networks (CNN), Long Short Term Memory (LSTM), Bidirectional LSTM, a few hybrid models, and various Transformer models.

3.2.1 Convolutional Neural Network - CNN

A Convolutional Neural Network (CNN) is a type of neural network that can analyze images or sequences that are designed in a grid-like manner. It is made up of multiple layers, including convolutional layers, pooling layers, and fully connected layers, which are stacked together in a specific order to create a network. The essential part of a CNN is its convolutional layers, where elements like edges, textures, and shapes are extracted from the input image by applying filters. The output is then passed to the pooling layer. This layer pre-

serves the important information while shrinking the spatial size of the convolved feature. This aids in minimizing computational power and dimension during data processing. Average pooling and maximum pooling are the two different forms of pooling. One or more fully connected layers are then applied to the output of the pooling layers in order to make a prediction or classify the image. [29]

CNNs are primarily associated with image processing, but when the input is represented as numerical vectors, they can also be used for text classification. CNN is very good at identifying hierarchies and local patterns in the text data. The only difference is that it uses 1D convolutional layers for text data instead of the 2D layers that are used for images.

3.2.2 Long Short Term Memory - LSTM

Long Short-Term Memory (LSTM) is a type of Recurrent Neural Network (RNN) that is capable of learning long-term dependencies in sequential data like time series, text, and speech. Traditional RNNs cannot remember long-term dependencies since they only have one hidden state. For this reason, they suffer from the vanishing gradient problem. LSTM networks address this issue by employing memory cells, which are capable of storing data for a long time. They capture and remember the important context in the sequence. The input gate, forget gate, and output gate control this memory cell. These gates decide what information to add to, remove from, and output from the memory cell. [30]

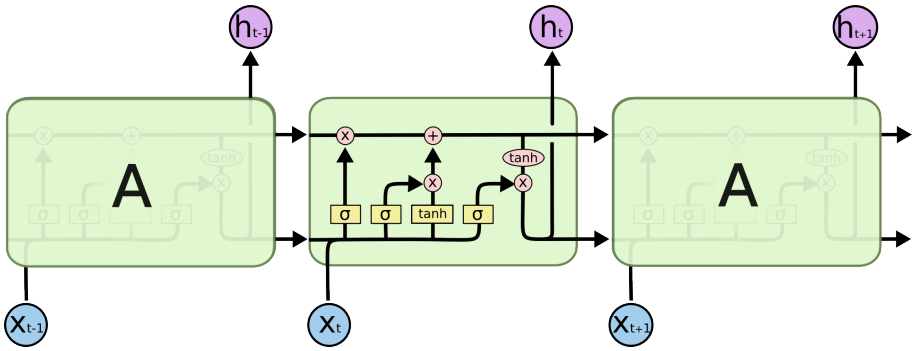

Figure 3.5: LSTM Network [31]

Forget gate: The forget gate either removes data that is no longer required in the cell state or saves data for later use by employing a sigmoid activation function.

Input gate: The input gate modifies the cell state with useful information. First, the sigmoid function is used to regulate the information. Then a tanh function is used to construct a vector. Finally, the information needed to generate an update is obtained by multiplying the values of the vector and the regulated values. In this way, it quantifies the importance of the new information carried by the input.

Output Gate: The output gate is responsible for extracting valuable information from the

current cell state as the output. Initially, a sigmoid layer determines which parts of the cell state will be output. Next, the cell state is processed through a tanh function and multiplied by the output of the sigmoid gate. This ensures that only the selected components are included in the output.

3.2.3 Bidirectional LSTM - Bi-LSTM

Bidirectional LSTM is a type of recurrent neural network (RNN) that is able to process the input sequence in a forward (past to future) or backward (future to past) direction simultaneously. However, LSTM makes predictions based on the previous context by processing the input sequence either forward or backward at a time. The bidirectional LSTM consists of two LSTM layers.

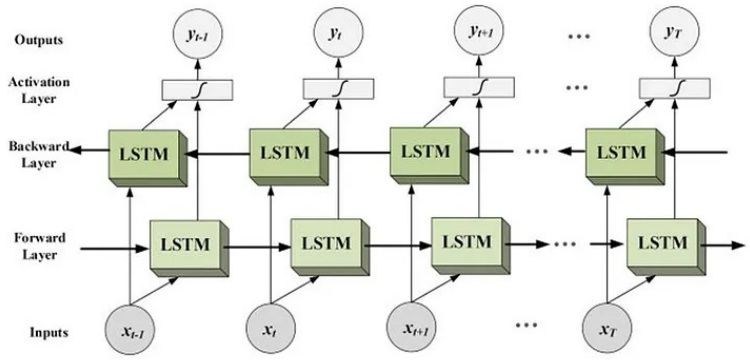

Figure 3.6: Bi-LSTM Network [32]

At each time step, the outputs of forward and backward LSTM are combined. For this reason, encoded words recognize the words that come before and after them. This makes it possible for the network to access data from the past and future simultaneously, enhancing its ability to recognize long-term dependencies and improve predictions in complex sequential data. [33]

3.2.4 BanglaBERT

A new era in Natural Language Processing (NLP) has been brought about by the emergence of pre-trained language models, which have made it possible to create advanced language models. Transformer-based models, such as BERT, are among the most often used models because of their exceptional efficiency. Unfortunately, the Bangla language's resource limitations have resulted in a lack of downstream task datasets and pre-trained language models. As a result, an improved BERT variation designed especially for the Bangla language is known as Bangla-BERT. This model has been optimized to match the distinct language

features of Bangla and achieves state-of-the-art performance in Bangla NLP tasks. Although Bangla-BERT shares the basic architecture of BERT, it integrates extra pre-processing processes to fit well into the large Bangla Corpora. This creative approach provides a useful tool for researchers and developers operating in this linguistic environment by addressing the unique requirements of the Bangla language in the NLP sector.

3.2.5 DISTIL BERT

In 2018, Google AI researchers released the BERT model. It was a fantastic work that brought a revolution in the NLP domain. However, the BERT model did have some drawbacks i.e. it was bulky and hence a little slow. In recent years, the transfer learning approaches in Natural Language Preprocessing with large pre-trained models have become mainstream. Model compression techniques could be leveraged to produce these massive/bulky models. [34]

A simplified version of the BERT model is the DistilBERT model. DistilBERT is a cost-effective pre-training model that is faster, lighter, and more compact. It has good performance results for on-device applications. Using knowledge distillation in the pre-training phase, a BERT model's size was reduced by 40% while maintaining 97% of its language understanding capabilities and increasing its speed by 60%. Knowledge distillation technique is also known as teacher-student learning. It's a model compression technique where a larger model (the teacher model) or an ensemble of models' capabilities are replicated by training a smaller model (the student model). By matching the output distribution, the student model is trained to generalize in the same manner as the teacher model. To use the inductive biases bigger models learned during pre-training, a triple loss is introduced. It combines distillation, cosine-distance losses, and language modeling. [34]

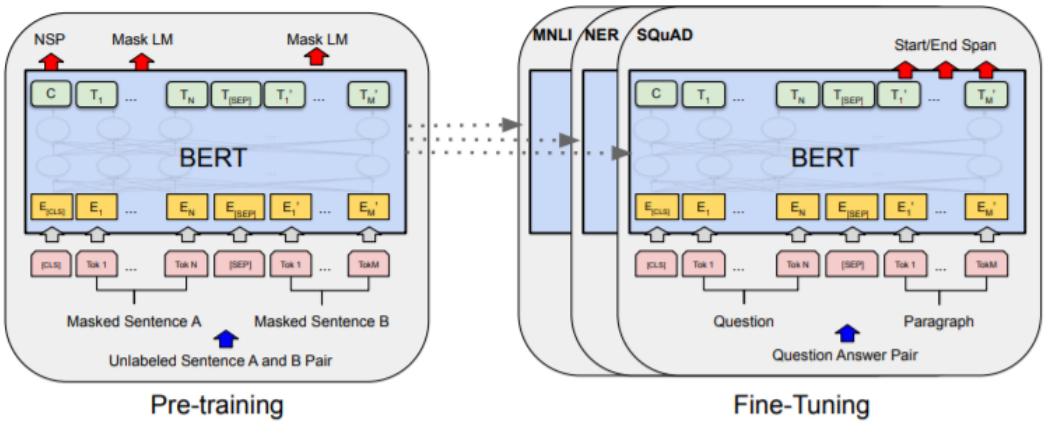

Figure 3.7: DistilBERT

3.2.6 XLM ROBERTA BASE

A multilingual version of RoBERTa is called XLM-RoBERTa. It has been pre-trained with 2.5TB of filtered CommonCrawl data comprising 100 languages. The transformers model RoBERTa was pretrained in a self-supervised manner on a large corpus. This indicates that it was trained using only the raw texts and an algorithmic procedure to create inputs and labels from those texts. No human labeling was done at all, which allows it to use a large amount of publicly available data. It was specifically pretrained with the Masked language modeling (MLM) objective. Using a sentence as an example, the model randomly masks 15% of the words in the input before running the entire masked sentence through the model, which must predict the masked words. This is in contrast to traditional recurrent neural networks (RNNs), which typically see words one after the other, or autoregressive models such as GPT, which internally mask future tokens. It allows the model to learn a bidirectional representation of the sentence. As a result, the model learns an inner representation of 100 languages, which can then be used to extract features useful for downstream tasks. [35]

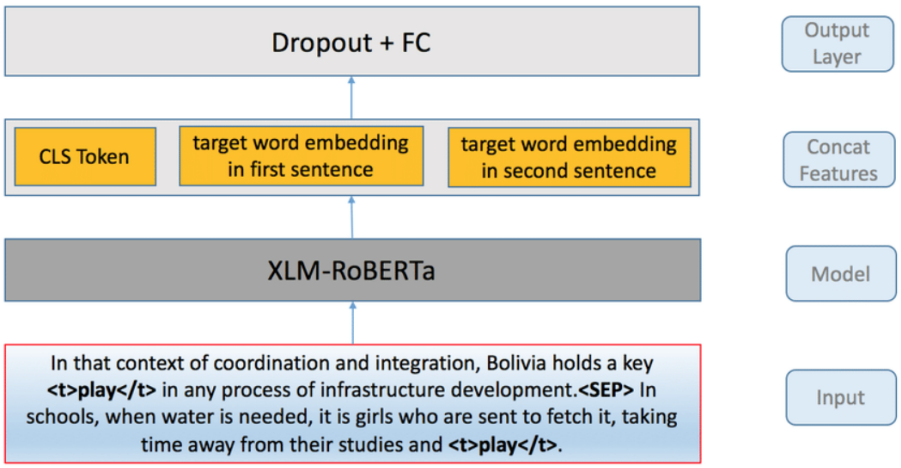

Figure 3.8: XLM ROBERTA BASE

3.2.7 IndicBERT

Indic-BERT is a multilingual ALBERT (A Lite BERT for Self-Supervised Learning of Language Representations) model pre-trained in 12 Indian languages and a recent derivative of BERT. [36] Assamese, Bengali, English, Gujarati, Hindi, Kannada, Malayalam, Marathi, Oriya, Punjabi, Tamil, and Telugu are the 12 languages that IndicBERT covers. Compared to other multilingual models (mBERT, XLM-R, etc.), IndicBERT has a lot fewer parameters and performs on par with or better than comparable models. [37]

From different language representations, Indic-BERT extracts the context in both ways. Indic-BERT is used to capture the linguistic and semantic aspects of a multilingual sen-

tence. YouTube posts and comments can contain multiple sentences. These multilingual input sentences can be combined into a single sequence for input representations by Indic-BERT. Token, segment, and positional embeddings are combined to create Indic-BERT embeddings. The addition of a classification layer at the bottom of the trained model allows it to be adjusted to the needs of the downstream jobs. For the objective of analyzing sentiment stated in social media, Indic-BERT might be used. [38]

The Indian subcontinent is home to many languages and dialects, this model is especially useful for applications like sentiment analysis, chatbots, content analysis, and language translation. Like previous BERT-based models, IndicBERT has the benefit of having been pre-trained on a large corpus of textual material, which enables it to extract rich semantic and contextual information that may be applied to a variety of natural language processing applications.

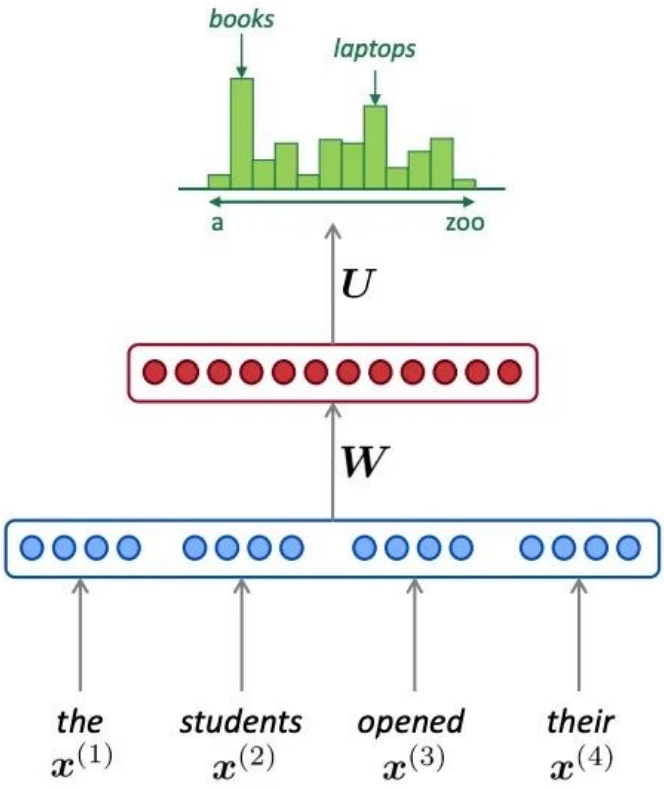

Figure 3.9: Indic-BERT

3.2.8 m-BERT

The multilingual version of the well-known BERT (Bidirectional Encoder Representations from Transformers) model is known as mBERT or Multilingual BERT. It is primarily trained on monolingual corpora across 104 languages. Consequently, mBERT is able to comprehend and be aware of the connections between words in all 104 languages simultaneously. This singular language model has remarkable proficiency in zero-shot cross-lingual model transfer—a process where task-specific annotations in one language are used to improve the model for evaluation in a different language.When content from various languages is semantically connected, MBERT recognises this.When dealing with data in, say Bangla, we can utilize the underlying MBERT model and refine it further with additional English-only content.The model will return high scoring, semantically related Bangla language content if you tell it what English language stuff is important to you.

It can produce and understand text in a variety of languages. Without the requirement for language-specific models, it can capture cross-lingual relationships and perform better on a variety of tasks thanks to this multilingual pre-training. For many languages, pre-trained word embeddings can be produced using mBERT. With the help of these embeddings, one can utilize multilingual text input for a variety of downstream NLP tasks without requiring a different pre-trained model for each language.

In an attempt to shed light on this, it offer a plethora of probing experiments that demonstrate: transfer can occur between languages with different scripts; it is most effective between typologically similar languages; monolingual corpora can train models for code-switching; and the model can identify translation pairs. Based on these findings, we may say that although M-BERT generates multilingual representations, these representations have systemic flaws that impact specific language pairs.

We employed the “BertForSequenceClassification” class from the Hugging Face Transformers library, using the pre-trained "bert-base-multilingual-cased" model. The preprocessing involves encoding sentences using the BERT tokenizer, ensuring that special tokens are added, and handling padding and truncation to a specified maximum length. The labels associated with each comment are encoded into numerical values using a predefined dictionary. Additionally, attention masks are created to distinguish between actual tokens and padding tokens in the encoded sequences. For optimization, the Adam optimizer is employed with a learning rate of 3e-5, epsilon of 1e-8, and weight decay of 0.01. These hyperparameters are commonly used in fine-tuning BERT models. The scheduler adjusts the learning rate during training, starting with a warm-up phase where the learning rate gradually increases from 0 to the specified rate, followed by a linear decay over the training steps. The forward and backward propagation are performed, and gradients are clipped to prevent exploding gradients. The training loss is accumulated and normalized by the number of batches. After

each epoch of training, the model is evaluated on a validation set [39]

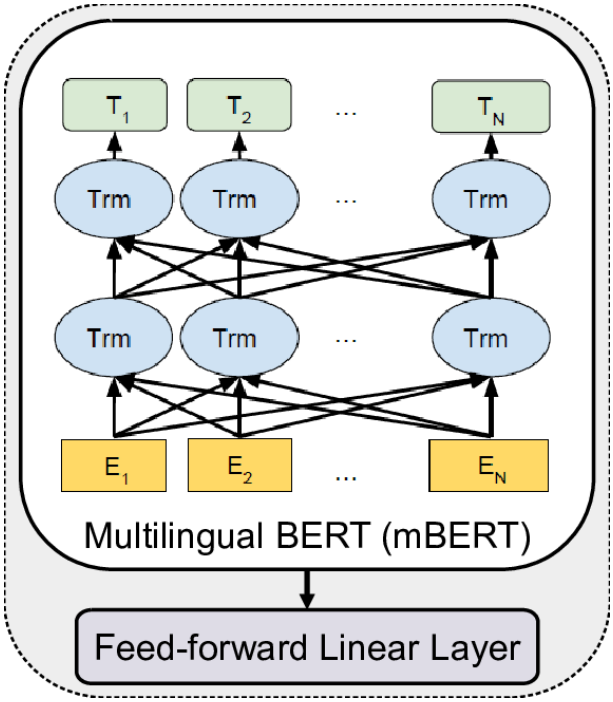

Figure 3.10: mBert [40]

3.2.9 Xlm Roberta Large

The multilingual RoBERTa version is called XLM-RoBERTa. On 2.5TB of filtered CommonCrawl data with 100 languages, it was pre-trained.

Facebook AI created a kind of pre-trained language model called XLM-Roberta. The BERT (Bidirectional Encoder Representations from Transformers) architecture serves as its foundation, and it is expanded to support numerous languages. The abbreviation "XLM" stands for "Cross-Lingual Model," indicating the model's multilingual understanding and generation capabilities.

Being a larger variant of the XLM-Roberta model, XLM-Roberta Large probably includes more parameters than its smaller counterparts. This could lead to improved performance on some tasks involving natural language processing.

RoBERTa is a transformers model that was pretrained in a self-supervised manner on a sizable corpus. This indicates that it was pre trained using only the raw texts and an automatic process to create inputs and labels from those texts. No human labeling was done at all, which allows it to use a large amount of publicly available data. To be more exact, the Masked language modeling (MLM) objective was used during pre-trained training. The model takes a sentence and masks 15% of the words in the input at random. The model then runs the entire masked sentence through the model and needs to predict the words that are

masked. This is not like autoregressive models like GPT, which internally mask the future tokens, or conventional recurrent neural networks (RNNs), which typically see the words one after the other. It enables the model to pick up a sentence's bidirectional representation. In this way, the model learns an inner representation of 100 languages, from which it can extract features useful for downstream tasks. For example, if you have a dataset of labeled sentences, you can use the features generated by the XLM-RoBERTa model as inputs to train a standard classifier. [41]

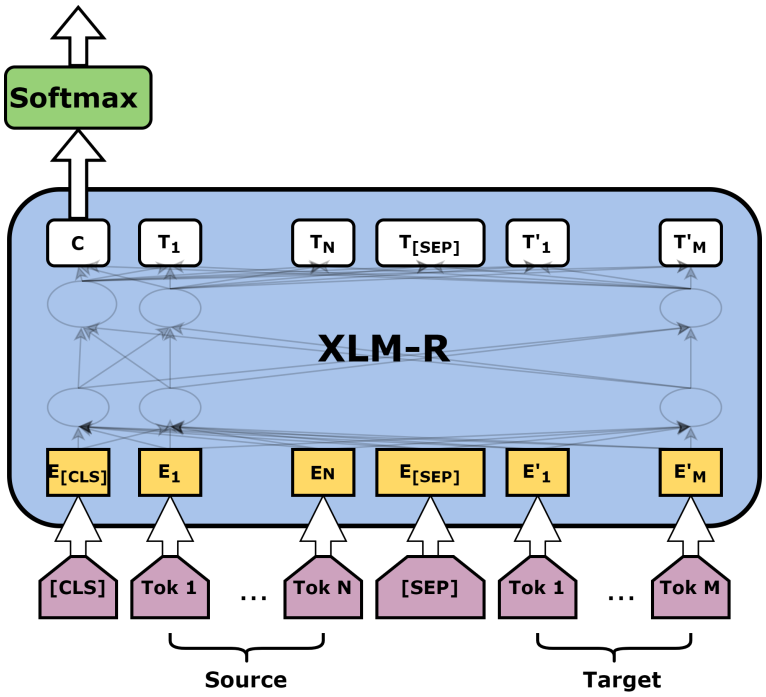

Figure 3.11: XLM-RoBERTa Large

Chapter 4

Methodology

4.1 Methodology

Machine learning and deep learning algorithms have become increasingly important in the field of cyberbullying detection due to the large amount of unstructured data that needs to be processed to detect cyberbullying incidents. These algorithms can help identify patterns, relationships, and trends in the data that may be difficult to identify using traditional methods.

4.2 Workflow Strategy

In order to complete our goal of detecting cyberbullying texts, we have followed a set of sequential steps to conduct our research. Our top priority was obtaining a relevant dataset, after which we prepared it for model application by pre-processing, data cleaning, and feature extraction. Our strategy included the use of both deep learning and machine learning models. We used these models on balanced and unbalanced datasets to investigate multi-class and binary classification scenarios.

Using a chart that is provided below, a pictorial representation of the process that we followed are described in Figure 3.1:

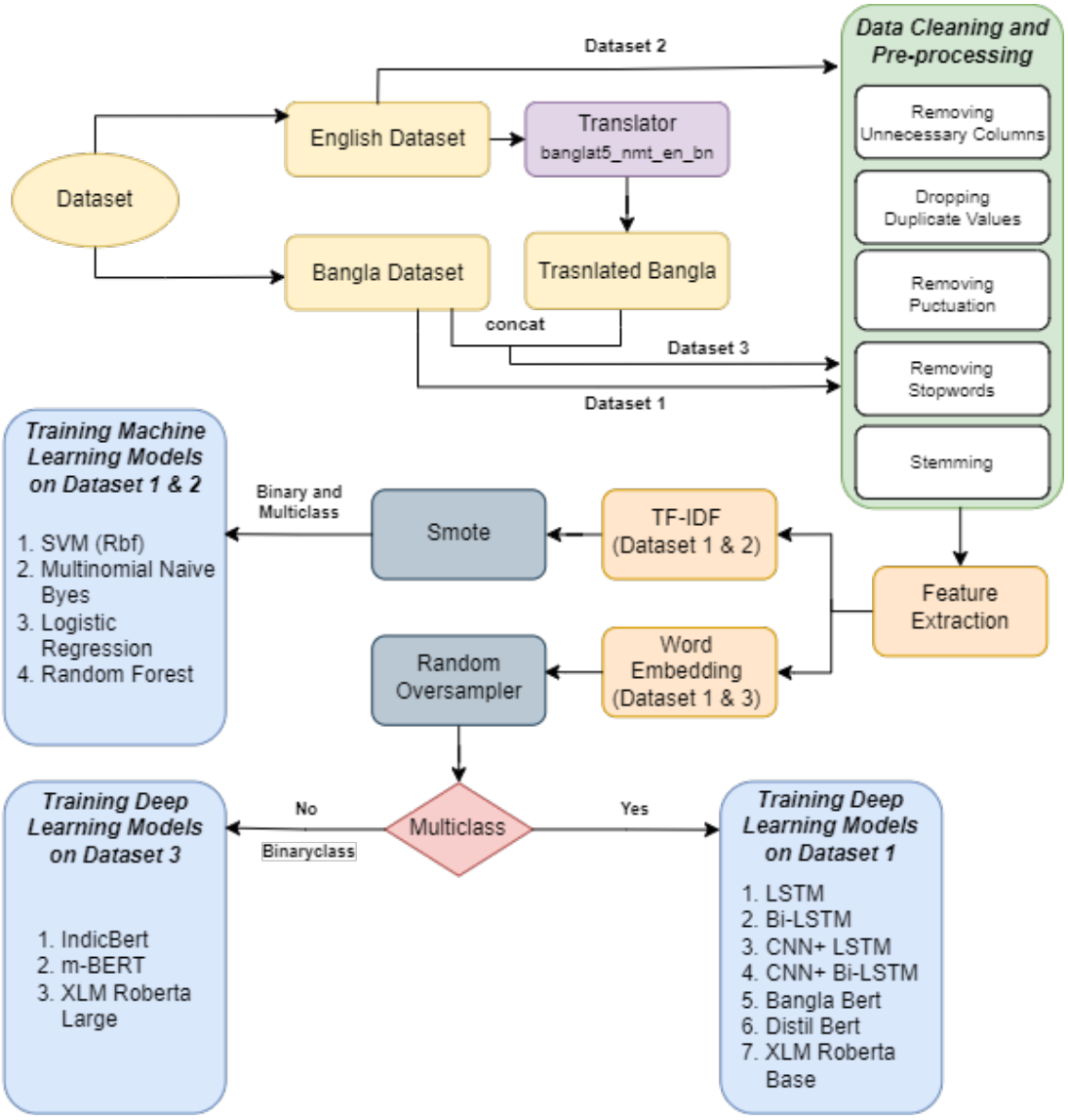

Figure 4.1: Work Flow Diagram.

4.3 Data Collection

Data collection is an essential part of identifying cyberbullying since it serves as the foundation for effective detection algorithms. Without appropriate and relevant data, it is difficult to train and evaluate machine learning or deep learning models for cyberbullying detection accurately. Data collection in cyberbullying detection involves collecting and labeling datasets that include various forms of cyberbullying. The essential component for creating accurate models of cyberbullying is an appropriate dataset. To prevent biases, the dataset needs to be large, diverse, and balanced, that extend a wide range of features and languages. The most critical objective of data collection is to ensure that accurate decisions can be made for research and that the data is reliable and information-rich for statistical analysis. Fur-

thermore, gathering data enables us to detect changes in language usage and cyberbullying patterns, which can increase the precision of detection models. Therefore, data collection helps in identifying new kinds of cyberbullying, which can be useful for updating existing detection models or creating new ones.

4.4 Dataset

For our English cyberbullying detection, we need to gather a large amount of data as training data for the cyberbullying detection algorithm. For this purpose, we used Daviston's Cyberbullying Classification dataset from Git Hub, which contains data about 24,783 tweets. [42] The dataset is labeled according to the class of cyberbullying:

- Cyberbullying
- Not Cyberbullying

The data of each label has been balanced in order to contain 20620 of bullying class, and 4163 of Not bullying class. Figure 2.1 shows the ratio of label data to provide details of each class.

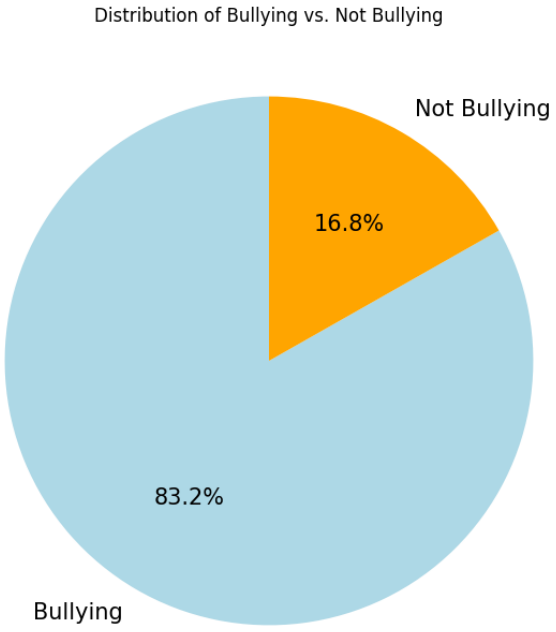

Figure 4.2: English Dataset labels ratio

Figure 4.2 shows labels around 83.2% of data is Cyberbullying data, and 16.2% is Not Cyberbullying data.

For our Bangla dataset [43], we have a significant amount of data that we have collected to identify cyberbullying in Bangla. There have been 44001 comments collected in total. With the help of natural language processing, the dataset seeks to determine if a statement is a bully expression or not and, if it is, to what extent it is unacceptable. With the aid of professionals and general agreement, the comments are classified into various types of harassment. There are five classifications in our Bangla dataset—sexual, not bullying, troll, religious, and threat—with respective incident counts of 8928, 15340, 10462, 7577, and 1694. The following bar chart illustrates it:

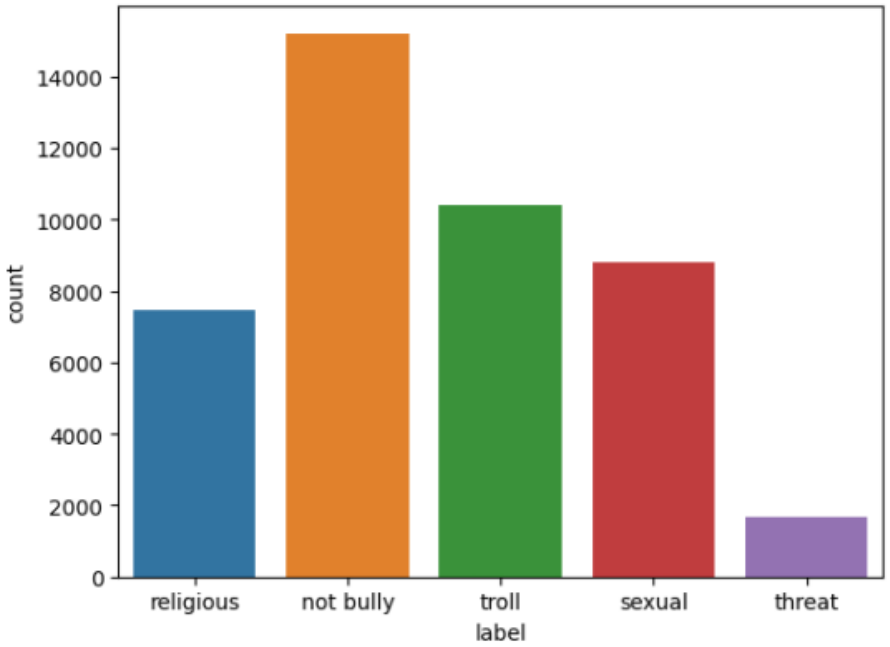

Figure 4.3: Bangla Dataset labels ratio

4.5 SMOTE

Synthetic Minority Over-sampling Technique, or SMOTE, is a widely used method in the fields of data imbalance. It is specially made to deal with the problem of datasets that are unbalanced, meaning that one class is much underweight in comparison to the others. Unbalanced datasets can create biasness in a model. As a result, models do not work properly for unseen data. To balance the class distribution, SMOTE creates synthetic samples for the minority class. By interpolating between existing minority class instances, the method finds instances of the minority class and generates synthetic examples. Giving the model more representative training data for the minority class helps to increase the accuracy with which it can classify instances belonging to that class.

4.6 Random Oversampling

Random Oversampling is a technique employed in imbalanced datasets When one class is greatly outnumbered in an unbalanced dataset. The method involves duplicating instances from the minority to correct for the imbalance. The main objective is to increase the machine learning model's sample size of the minority class in order to facilitate better learning and possibly enhance performance.But there are things to keep in mind, like the possibility of overfitting, particularly when working with small datasets. Random Oversampling is one of several approaches to handling class imbalance.

4.7 Data Preprocessing

Data preprocessing is an essential step in cyberbullying detection that involves cleaning and transforming raw text data into a structured format. Transforming unstructured text data into a structured format can be used as the goal of data preparation. Most often, text data contains irrelevant information, such as HTML tags, special characters, and punctuation marks, which include noise and unimportant information. Preprocessing helps to remove these unwanted elements to ensure that the remaining text data is relevant and meaningful. Text normalization, which includes changing the text's data to lowercase, getting rid of unnecessary words, and stemming or lemmatizing words to get to their root form, requires preprocessing. Data preprocessing helps to reduce the complexity of the text data, which helps in feature extraction, and model training to improve the accuracy and effectiveness of the prediction models for cyberbullying detection.

The data preprocessing steps are done in the following ways:

- **Lowercasing:** Lowercasing involves converting all text data to lowercase. This technique is useful because it standardizes the text [44]. When text data is not standardized, it can lead to duplicate entries. For example, "Bad" and "bad" would be treated as two different words if lowercase didn't apply to them. It would be helpful to remove potential sources of noise from the data. For our English dataset, we have used the text lowercasing process.

  Sample Input: 'He',' Rahim', 'Beautiful'
  Sample Output: 'he','rahim', 'beautiful'

- **Stop Word Removal:** The removal of a commonly occurring collection of terms that do not add much meaning to a sentence is the removal of stopwords. Figure 3.3 shows an example of removing stop words.

Sample input: ‘আমি’, ‘ওর’, ‘হাত’, ‘ভেঙে’, ‘দেই’

(‘I’, ‘break’, ‘his’, ‘arm’)

Sample output: ‘হাত’, ‘ভেঙে’, ‘দেই’

(‘break’, ‘arm’)

Figure 4.4: Removal of Stopwords [9]

- **Removing special characters and punctuation:** Punctuation and special characters can increase text information that may not be necessary for classification tasks. Removing them may help improve the accuracy of the prediction models.

  Sample Input: Hello!!!, World!!%@.
  Sample Output: Hello World

- **Lemmatization:** Lemmatization is a useful technique that is used to reduce words to their base or dictionary form. Additionally, it helps with the text's elimination of unnecessary details. For example, the word "ran" may be lemmatized to "run," "running" to "run," "wolves" to "wolf," and so on.

- **Translating Emoticons:** Emojis and emoticons are commonly used in text corpora and text data, particularly in the text that has been gathered from social media. However, if they are not translated into equivalent text, these emojis are essentially meaningless in the context of text analysis. Many works related to cyberbullying and sentiment analysis have removed emoticons while preprocessing their data. However, translating emoticons and using their meaning will improve text data comprehension. For this reason, we have used a Python module that allows the conversion of all emoticons into Bangla text.

- **Tokenization:** The first step in natural language processing is tokenization.It is a crucial preprocessing stage that enables models to comprehend and interpret the underlying structure of textual data in both machine learning and deep learning models. Depending on the particular task and requirements, these tokens may be single words, sentences, or even individual characters. However, Both traditional machine learning models and advanced transformer models start with tokenization.

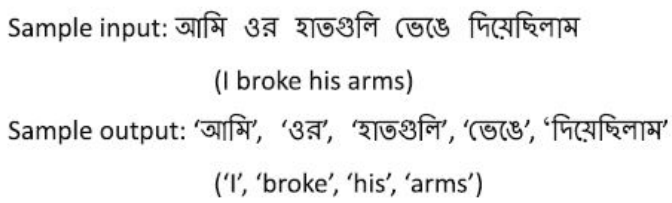

Figure 4.5: Tokenization of Bangla Sentence [9]

- **Stemming:** Stemming is the process of removing the root word from the given word. The fundamental idea of stemming is to reduce complex grammar and word structures to their base, stem, or root form.

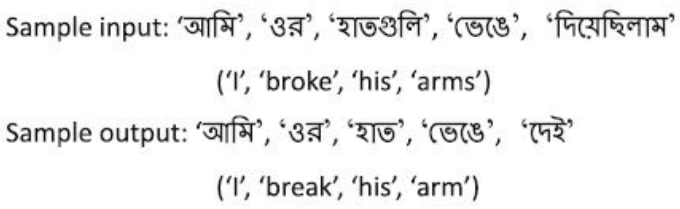

Figure 4.6: Stemming of Bangla Sentence [9]

4.8 Feature Extraction

Over the past few years, more people have been using social media sites like Facebook, WhatsApp, Twitter, and Instagram. Through these sites, people exchange a huge amount of information, some of which use offensive language. Features in machine learning and data analysis refer to the parameters or characteristics that are used to build predictive models. Data analysis, prediction, and classification are dependent on these features. As cyberworld datasets are created with various properties, it becomes more challenging to select features for prediction. The quality of a dataset can be increased by optimizing its features Hence, feature extraction is important since it helps to define complex data sets with fewer characteristics. The different data types of features are used for cyberbullying prediction. Feature extraction methods implemented on any dataset depend on the data type. The majority of cyberbullying incidents seem to involve negativity. Special characteristics, such as sexuality, politics, and race, can also be used to predict labels. Feature extraction is useful when there is a need to reduce the amount of redundant data without losing important or relevant information.

4.9 TF-IDF

For feature extraction, we have used TF-IDF. The abbreviation TF-IDF stands for "Term Frequency-Inverse Document Frequency." It is a metric for quantifying the significance of a phrase inside a document or corpus of documents that is used in information retrieval and text mining. By combining the TF and IDF values for a phrase in a document, TF-IDF is created. When compared to the overall number of terms in a document, TF shows how frequently a specific term appears. IDF (Inverse Document Frequency) measures a term's significance in a corpus by taking into account how frequently it appears in various documents. IDF emphasizes terms that are found in fewer documents and lessens the weight of terms that are found in many documents. Here is the formula for how the TF-IDF works

$$TF(t,d) = \frac{\text{number of times t appears in d}}{\text{total number of terms in d}} \tag{4.1}$$

$$IDF(t) = log(N/1 + df) \tag{4.2}$$

$$TF - IDF(t,d) = Tf(t,d) * IDF(t) \tag{4.3}$$

With the use of this TF-IDF, we can turn our text into numerical data for machine learning.

4.10 Word Embedding

Words in the input sentence must be represented as numerical vectors in order to carry out any kind of machine learning or deep learning task. Each word in a vocabulary is mapped to a vector, where words with similar meanings or contexts have similar vector representations. Word embeddings aim to capture not just the syntactic and semantic aspects of the paragraph but also its context and similarities. It also solves the issue of sparse matrix encountered in techniques like Bag of Words (BOW), CountVectorizer, and TFIDF because these methods rely on word counts in a sentence. Word2Vec is one of the most popular techniques for word embedding. Its input is a text corpus and produces vector representations of words in its vocabulary. Word2vec aims to capture semantic relationships between words and their context by clustering vectors of related words together in vector space.

4.11 Categorical Encoding

The process of converting categorical data to numerical values so that a machine learning algorithm can understand it is called Categorical Encoding. There are two ways to convert them: label encoding and one-hot encoding. [45]

- **Label Encoding:** Label Encoding is used to represent each category of a dataset using a unique integer value. This type of encoding can help to preserve the meaning of the element as the values are assigned sequentially. **Example:** The dataset that we are using classifies the bully texts using five labels. They are: "Not Bully," "Religious", “Sexual”, "Threat", “Troll”. After label encoding, the labels might be represented as 0, 1, 2, 3, 4 respectively.

- **One-hot encoding:** One-hot encoding is utilized in the classification task to encode categorical labels into a numerical representation that is useful for training and prediction. It converts class labels to binary vectors, where a binary bit represents each category. The true labels are represented as 1, whereas all other representations are set to 0. The lengths of the binary vector are equal to the number of labels.

4.12 Translation

For translating text from English to Bangla, we’ve used the banglat5_nmt_en_bn pre-trained model to translate our English corpus into Bangla and add it later to the Bangla dataset. The pre-trained model uses a specific normalization pipeline to make sure to normalize the text units before tokenizing them for optimal outcomes. [46] In this work, we develop a customized sentence segmenter for Bengali and suggest two new approaches—batch filtering and aligner ensembling—for simultaneous corpus building on low-resource setups. By combining the two approaches with the segmenter, BUET CSE NLP creates a high-quality parallel corpus of Bengali-English sentences that has 2.75 million sentences total, over 2 million of which were previously unavailable. By utilizing neural models for training, we surpass earlier methods in Bengali-English machine translation by almost 9 BLEU scores. [47]

4.13 Model training and performance analysis

Model training and performance analysis are two of the most crucial phases in machine learning (ML) and deep learning (DL) lifecycles. To optimize a model’s performance during training, a number of procedures must be followed, including data cleaning, selection of

optimizers, loss functions, and hyperparameter tuning. These procedures are necessary to ensure that the model is properly trained on a dataset, so it can generalize to new data and functions satisfactorily. It's crucial to keep an eye on the model's performance following training. Performance analysis uses evaluation criteria such as recall, accuracy, precision, f1-score, confusion matrix, learning curves, etc. The model is deployed they have demonstrated sufficient performance.

4.14 Cost Analysis

Our thesis expenditures depended on some parameters which are described below:

In order to complete our study, we had to execute our code in Collab. For machine learning, this was simple; however, for deep learning and translation, we needed to work with large amounts of data, which required us to purchase a Google Collab Pro subscription for 1108 Taka per month. For distribution, we had to print several copies of our thesis pre-defense report, and we will need to print further copies for the thesis defense and final submission in the future. We calculated that this would cost 5000 Taka. The internet is the most significant medium; it was the foundation of our entire thesis project. The internet was required for the following tasks: reading research papers, gathering datasets, creating thesis reports in Latex, running code in collab, communicating on Zoom or Google Meet, and taking courses. We calculated a cost of an average of 5000 Taka for each. So, the total cost for our study comes to 26,108 Taka at the end. Table 4.1 shows the expenses in a tabular format:

Table 4.1: Cost Analysis Table

Expense Type	**Cost (BDT)**
Purchasing Google Collab Pro	1108
Printing and Binding	5000
Purchasing Broadband Internet Connection	20000
Total Cost	**26108**

4.15 Gantt Chart

The Gantt chart depicted in Figure 4.7 illustrates our workflow for individual tasks completed during a specified duration for our thesis project, which spanned from October 26, 2022, to November 21, 2023.

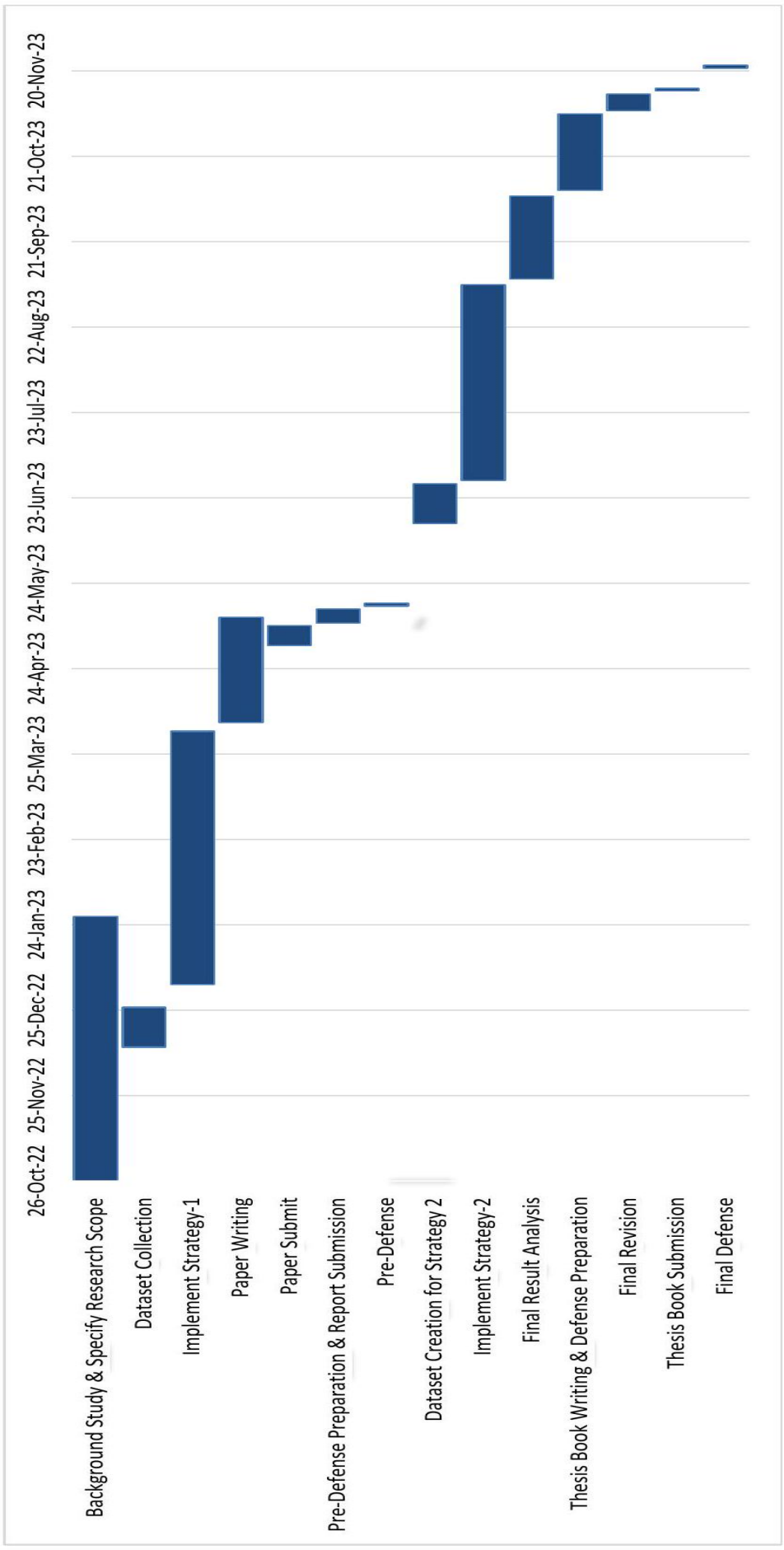

Figure 4.7: Gantt Chart

Chapter 5

Result Analysis

After preprocessing, we split the dataset into train and test data. Once the dataset have been divided into train and test data, the next step was to use machine learning models on the train data. For Cyberbullying classification, we have used SVM, Naive Bayes, Random Forest Classifier, and Logistic Regression.

5.1 Machine learning model performance

We have measured Accuracy, Precision, Recall, and F1-score for evaluating the performance of classification models. Each of these metrics measures provides valuable information for understanding the model's strengths and weaknesses.

5.1.1 Support Vector Machine

We have used linear SVM to train our model for the English dataset and for the Bangla dataset, we have used Radial Basis Function(RBF) SVM and have evaluated the performance of test data for bullying and not-bullying classification. SVM(RBF) also have been used for multi-class classification. The performance of SVM for the binary classification for both languages is shown in Table 5.1 below,

Table 5.1: Performance comparison of SVM for binary classification

Label	Bangla			English		
	Precision	Recall	F1-score	Precision	Recall	F1-score
Bully	0.84	0.77	0.80	0.97	0.97	0.97
Not Bully	0.79	0.85	0.82	0.86	0.86	0.86
Overall	**Accuracy: 0.81**			**Accuracy: 0.95**		

In our model, we have plotted a confusion matrix for binary classification that represents the model's performance for the English language dataset. Figure 5.1, shows the confusion matrix of the English dataset for binary classification.

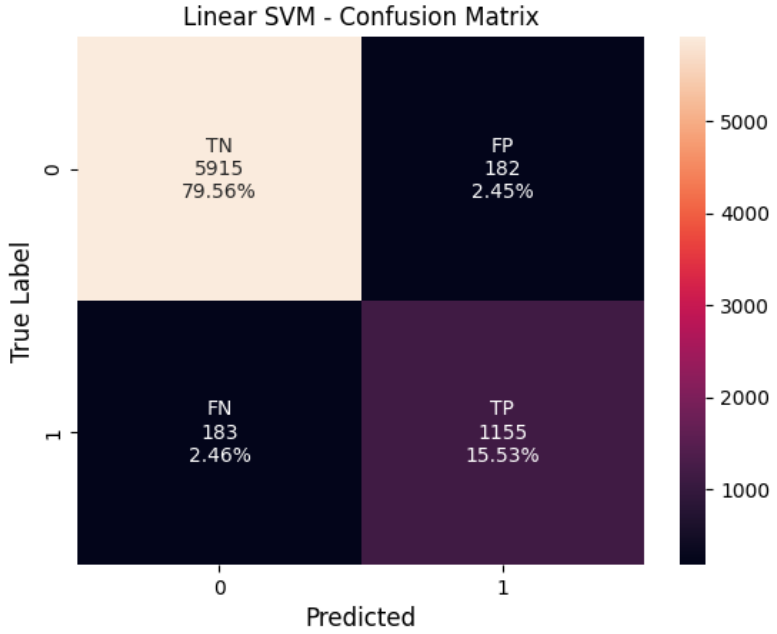

Figure 5.1: Binary classification for English Dataset

We have measured the performance of the multi-class classification for our Bangla dataset. The multi-class classification of SVM for the Bangla dataset is shown below in Table 5.2,

Table 5.2: Evaluation Metrics of SVM for multi-class classification (Bangla Dataset)

Label	**Precision**	**Recall**	**F1-score**
Not Bully	0.69	0.68	0.68
Religious	0.66	0.61	0.63
Sexual	0.69	0.53	0.60
Threat	0.60	0.94	0.73
Troll	0.69	0.52	0.59
Overall	**Accuracy: 0.66**		

Here below in Figure 5.2, shows the plots which represent binary classification for both Bangla and English datasets and multi-class classification for the Bangla dataset,

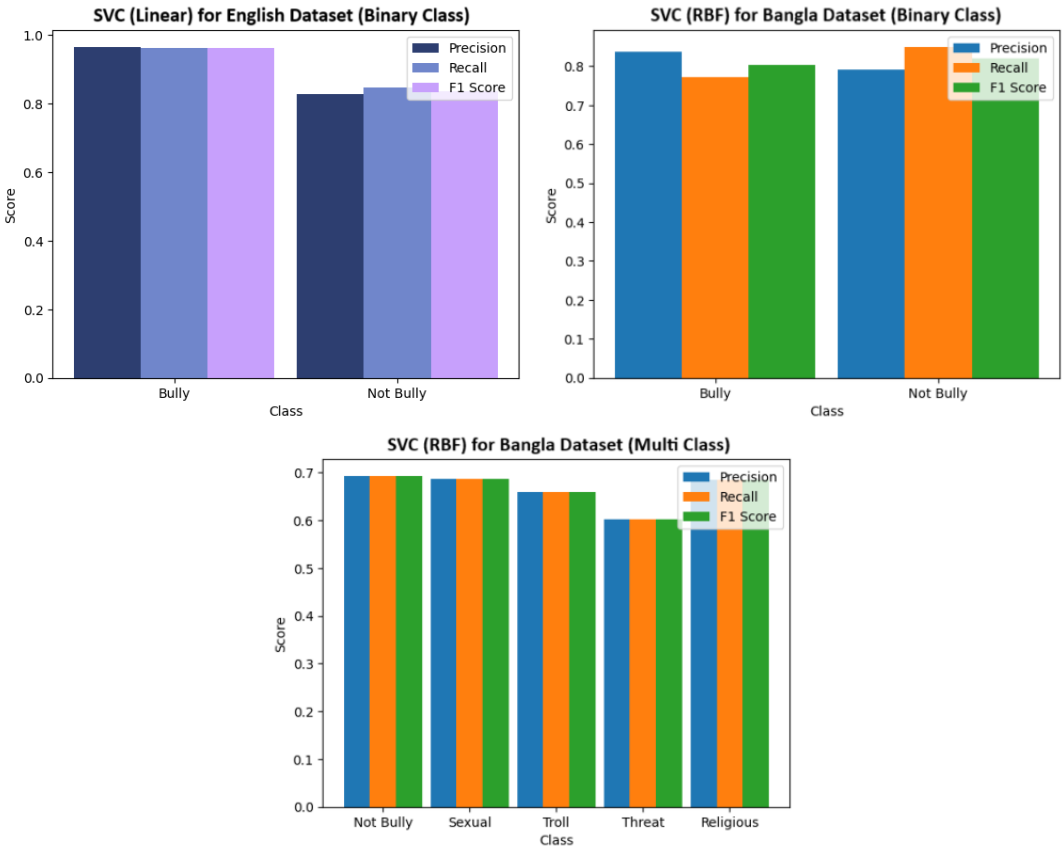

Figure 5.2: Binary and multi-class classification for English and Bangla Dataset

So we can see that, for binary classification, by using SVM we have got an accuracy of 95% for our English dataset, and for the Bangla dataset, we have got an accuracy of 81%. For multi-class classification on the Bangla dataset, we have got an accuracy of 66%.

5.1.2 Naïve Bayes

Naive Bayes is a probabilistic machine learning algorithm that is often used for classification problems. As Multinomial Naive Bayes is one of the most popular supervised learning classifications that is used for the analysis of the categorical text data [48], we have used the Multinomial Naive Bayes to train our model and evaluate the performance of the model. The performance of Naive Bayes is shown in Table 5.3 below,

Table 5.3: Performance comparison of Multinomial NB for binary classification

Label	Bangla			English		
	Precision	Recall	F1-score	Precision	Recall	F1-score
Bully	0.73	0.84	0.78	0.83	1.00	0.91
Not Bully	0.81	0.68	0.74	0.98	0.08	0.14
Overall	**Accuracy: 0.76**			**Accuracy: 0.83**		

In our model, we have plotted a confusion matrix for binary classification that represents the model's performance for the English language dataset. Figure 5.3, shows the confusion matrix of the English dataset for binary classification.

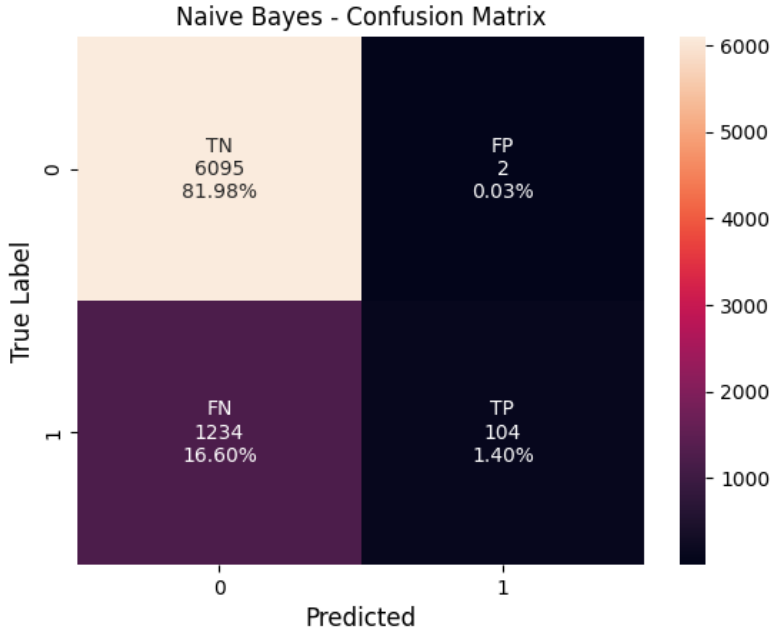

Figure 5.3: Binary classification for English Dataset

We have measured the performance of the multi-class classification for our Bangla dataset. The performance of the multi-class classification of Naive Bayes for the Bangla dataset is shown below in Table 5.4,

Table 5.4: Evaluation Metrics for multi-class classification (Bangla Dataset)

Label	**Precision**	**Recall**	**F1-score**
Not Bully	0.62	0.55	0.58
Religious	0.38	0.69	0.49
Sexual	0.51	0.42	0.46
Threat	0.59	0.45	0.51
Troll	0.56	0.42	0.48
Overall	**Accuracy: 0.50**		

Here below in Figure 5.4, shows the plots which represent binary classification for both Bangla and English datasets and multi-class classification for the Bangla dataset,

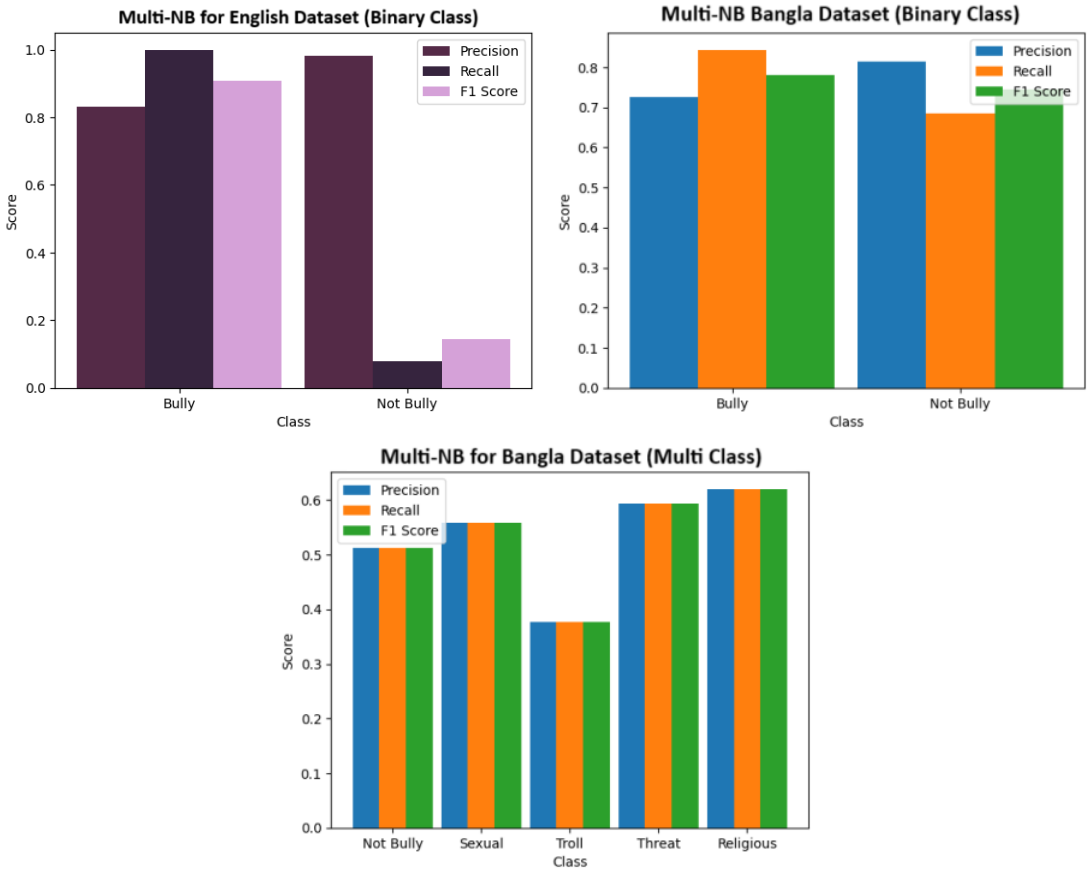

Figure 5.4: Binary and multi-class classification for English and Bangla Dataset

By using Naive Bayes we can see that, for binary classification, we have got an accuracy of 83% for our English dataset, and for the Bangla dataset, we have got an accuracy of 76%. For multi-class classification on the Bangla dataset, we have got an accuracy of 50%.

5.1.3 Logistic Regression

Logistic regression is a popular machine learning algorithm used for binary classification problems. It is a statistical model that estimates the probability of an instance belonging to a particular class based on its features. We have used Logistic Regression to train our model and evaluate the performance of test data for bullying and not-bullying classification. The performance of Logistic Regression is shown in Table 5.5 below,

Table 5.5: Performance of Logistic Regression for binary classification

Label	Bangla			English		
	Precision	Recall	F1-score	Precision	Recall	F1-score
Bully	0.78	0.73	0.76	0.93	0.98	0.96
Not Bully	0.75	0.80	0.77	0.90	0.68	0.77
Overall	**Accuracy: 0.77**			**Accuracy: 0.93**		

In our model, we have plotted a confusion matrix for binary classification that represents the model's performance for the English language dataset. Figure 5.5, shows the confusion matrix of the English dataset for binary classification.

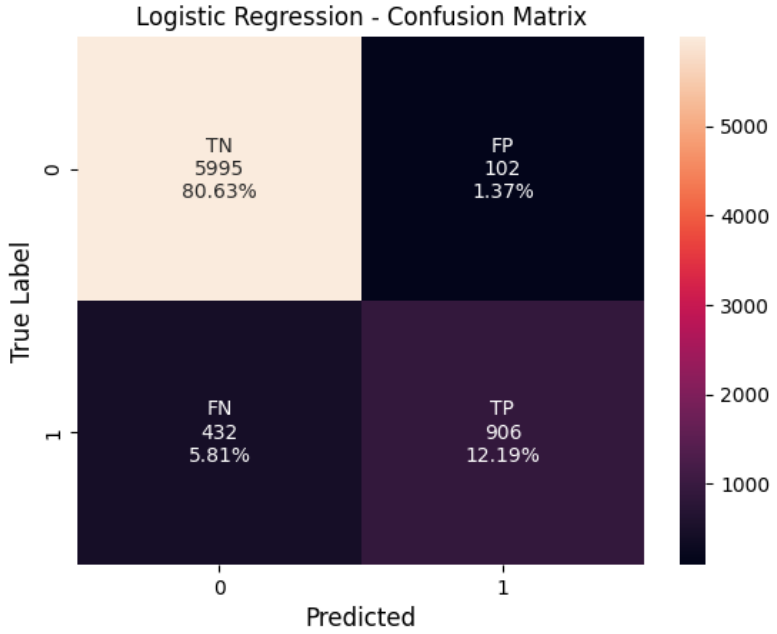

Figure 5.5: Binary classification for English Dataset

We have measured the performance of the multi-class classification for our Bangla dataset. The multi-class classification of Logistic Regression for the Bangla dataset is shown below in Table 5.6,

Table 5.6: Evaluation Metrics for multi-class classification (Bangla Dataset)

Label	**Precision**	**Recall**	**F1-score**
Not Bully	0.63	0.59	0.61
Religious	0.49	0.52	0.51
Sexual	0.53	0.42	0.47
Threat	0.48	0.72	0.58
Troll	0.58	0.43	0.49
Overall	**Accuracy: 0.54**		

Here below in Figure 5.6 shows the plots that represent binary classification for both Bangla and English datasets and multi-class classification for the Bangla dataset.

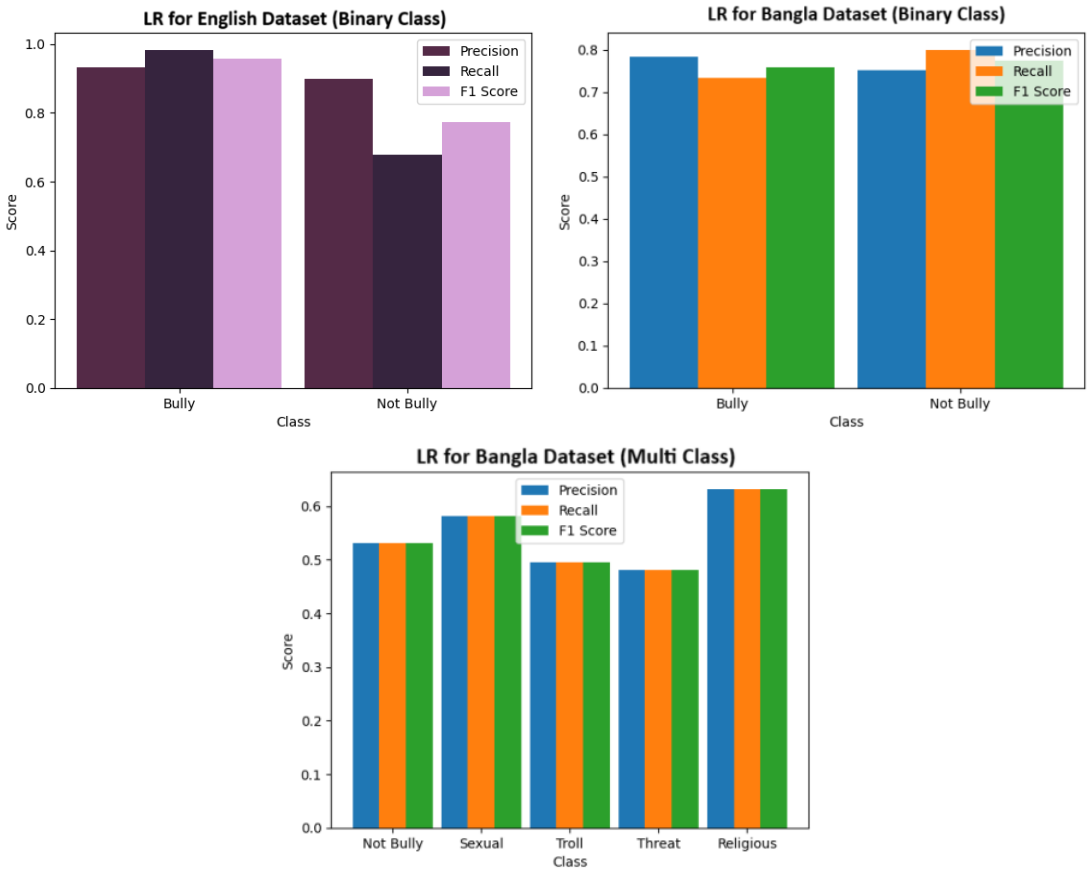

Figure 5.6: Binary and multi-class classification for English and Bangla Dataset

By using Logistic Regression for binary classification, we have got an accuracy of 93% for our English dataset, and for the Bangla dataset, we got an accuracy of 77%. For multi-class classification on the Bangla dataset, we have got an accuracy of 54%.

5.1.4 Random Forest

Random forest is a commonly-used machine learning algorithm that combines the output of multiple decision trees to reach a single result [49]. We have used the Random Forest classification algorithm to train our model for the English dataset and the Bangla dataset. The performance of the Random Forest for the binary classification for both languages is shown in Table 5.7 below,

Table 5.7: Performance of Random Forest for binary classification

Label	Bangla			English		
	Precision	Recall	F1-score	Precision	Recall	F1-score
Bully	0.80	0.84	0.82	0.95	0.98	0.96
Not Bully	0.83	0.79	0.81	0.89	0.75	0.82
Overall	**Accuracy: 0.82**			**Accuracy: 0.94**		

In our model, we have plotted a confusion matrix for binary classification that represents the model's performance for the English language dataset. Figure 5.7, shows the confusion matrix of the English dataset for binary classification. We have measured the performance

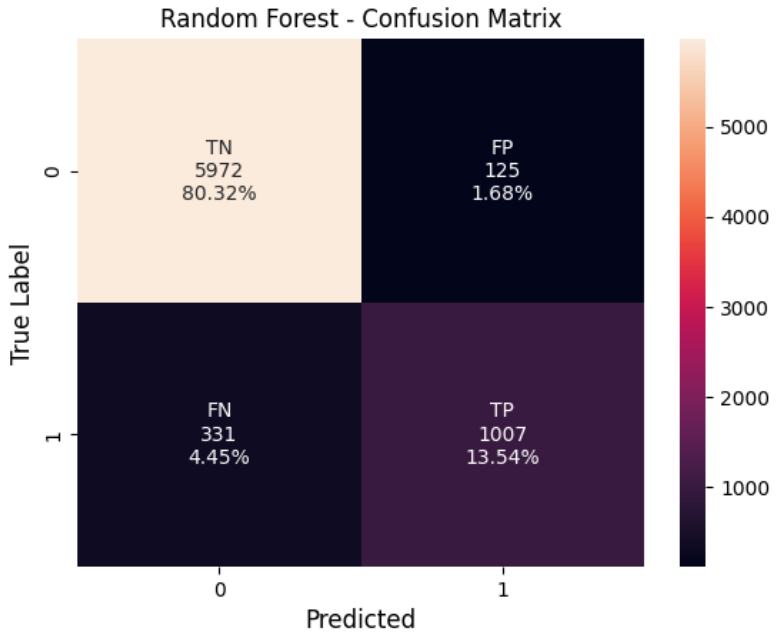

Figure 5.7: Binary classification for English Dataset

of the multi-class classification for our Bangla dataset. The multi-class classification of Random Forest for the Bangla dataset is shown in Table 5.8 below,

Table 5.8: Evaluation Metrics for multi-class classification (Bangla Dataset)

Label	**Precision**	**Recall**	**F1-score**
Not Bully	0.72	0.61	0.66
Religious	0.69	0.71	0.70
Sexual	0.72	0.59	0.65
Threat	0.63	0.95	0.76
Troll	0.69	0.56	0.62
Overall	**Accuracy: 0.68**		

Here below in Figure 5.8 shows the plots which represent binary classification for both English and Bangla datasets and multi-class classification for the Bangla dataset.

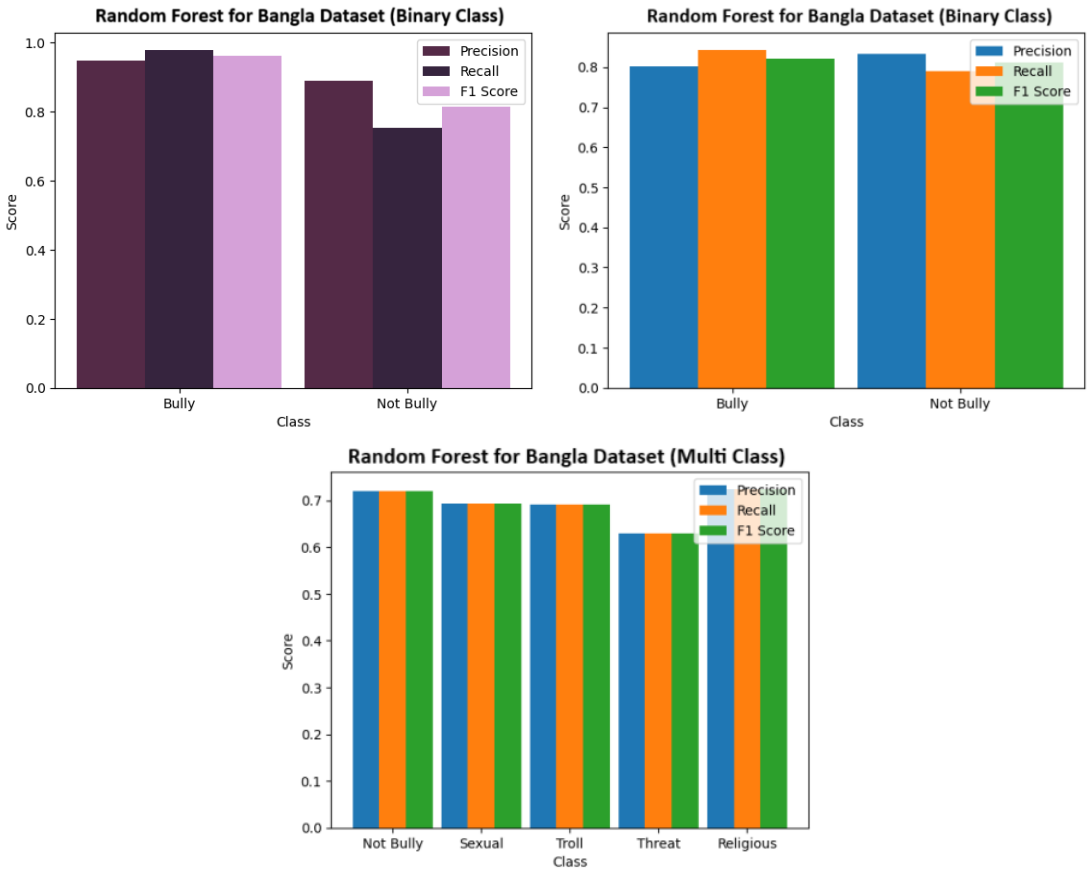

Figure 5.8: Binary and multi-class classification for English and Bangla Dataset

So we can see that, for binary classification, by using Random Forest we got an accuracy of 95% for our English dataset, and for the Bangla dataset, we got an accuracy of 81%. For multi-class classification on the Bangla dataset, we got an accuracy of 66%.

5.2 Deep learning model performance

As deep learning models, we used DistilBERT, XLM-RoBERTa Base, XLM-RoBERTa Large, Bangla BERT, IndicBERT, m-BERT, LSTM, Bi-LSTM, CNN+LSTM, and CNN+Bi-LSTM.

- For DisitilBERT, XLM-RoBERTa Base, and Bangla BERT, the corresponding learning rates, batch sizes, and epochs are 2e-5, 32, and 5. The split between training and testing was 90-10 and the split between training and validation data was 80-20. We used the Adam optimizer, a popular deep-learning optimization algorithm, to optimize the model's parameters for these specific models. Early-stopping criteria were used to avoid overfitting. This technique monitors the validation loss and stops training when there is no improvement, preventing the model from overfitting the training data.

- For LSTM, Bi-LSTM, CNN+Bi-LSTM, and CNN+LSTM, the dataset was initially divided

into 90–10 splits of train and test data. After that, a 75-25 split was applied to the training data to create training and validation sets. Before splitting, a word embedding model, such as Word2Vec from the Gensim library, was trained on the dataset. The texts were then tokenized for classification, which involved padding the sequences to a set length of 130 and transforming them into integer sequences and for the labels, a one-hot encoding method was utilized.

In order to ensure that the data flowed through each layer sequentially, every model architecture followed a sequential layer-by-layer methodology. For CNN+LSTM and CNN+Bi-LSTM, RMSprop was used as the optimizer having a learning rate of 0.001. The batch size was set to 64. On the other hand, for LSTM and Bi-LSTM, Adam was used as the optimizer having a learning rate of 0.0001, and the batch size was set to 128. All the models were trained for 30 epochs and the same early stopping and model checkpoints were used as callback functions to prevent overfitting. In order to manage the model's updates during training and measure training results, a suitable loss function, namely categorical cross-entropy loss was employed.

5.2.1 DistilBERT

We used the Hugging Face Transformers library, a powerful tool for working with pre-trained models, to perform text classification with DistilBERT. The "DistilBertTokenizer" was used and loaded with the "distilbert-base-multilingual-cased" pre-trained model. We tokenized the text with the DistilBertTokenizer and ensured that the maximum sequence length did not exceed 128 characters. We transformed the preprocessed dataset into a PyTorch dataset, which is required for it to work smoothly with the PyTorch framework. To measure the model's performance and control its updates during training, we used a suitable loss function for multi-class text classification, such as Cross-Entropy Loss.

We have worked with balanced data as well as unbalanced data. The results for both balanced and unbalanced data are given below-

- **Balanced Dataset**

Table 5.9: Performance metrics of DistilBERT (Balanced data)

Label	Precision	Recall	F1-score	Support
Not Bully	0.90	0.81	0.85	1450
Religious	0.95	0.95	0.95	1452
Sexual	0.90	0.93	0.91	1494
Threat	0.96	1.00	0.98	1459
Troll	0.84	0.85	0.84	1409

From performance metrics, we can see that, the threat label is working best among all other labels. Its support is 1459/7265 and the precision, recall, and f1-score are the best among all. Then comes sexual and religious. Which has also a very decent amount of good precision, recall, and f1-score. Not bully and troll label is also working very decent.

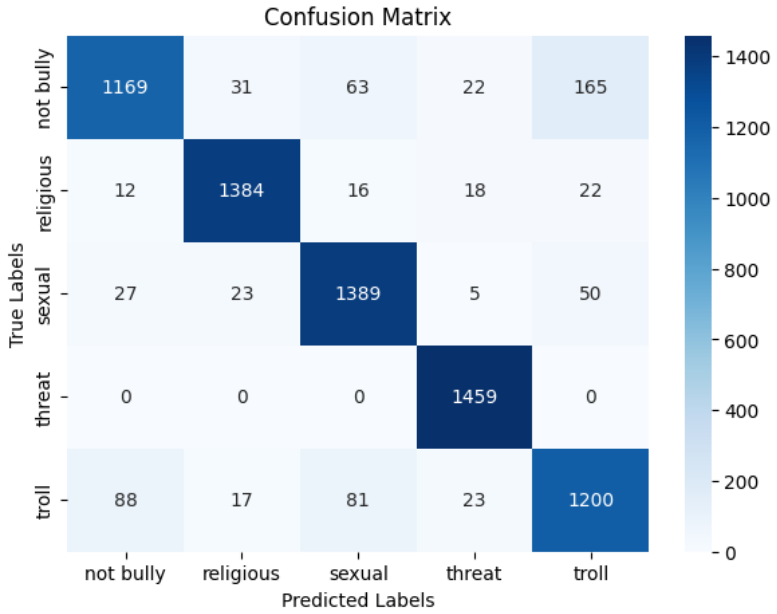

Figure 5.9: Heatmap of DistilBERT (Balanced data)

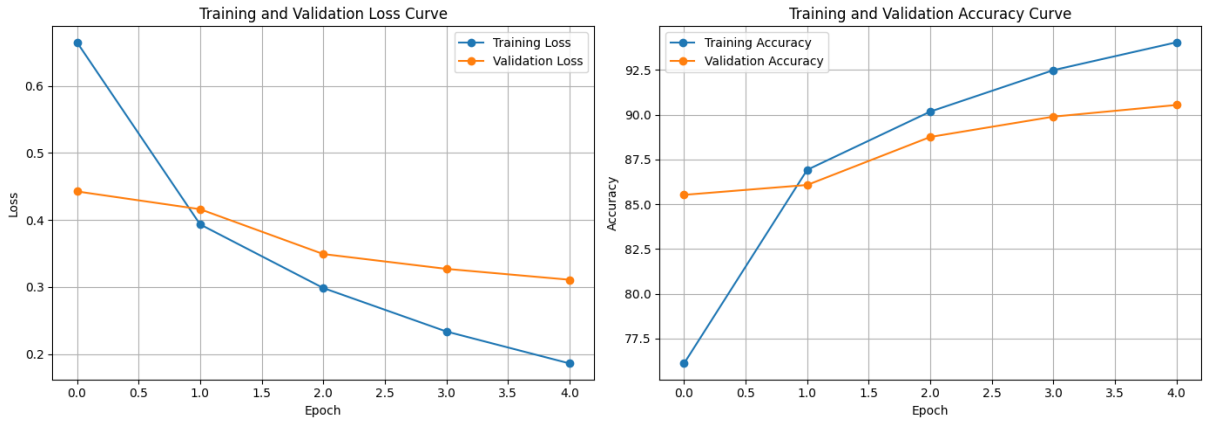

Figure 5.10: Loss and Accuracy Curve of DistilBERT (Balanced data)

From the above "Loss Curve" graph we can see that the blue marker indicates training loss and the orange marker indicates validation loss. After each epoch, we can see that the training loss is decreasing as well as the validation loss is also decreasing and the gap between training loss and validation loss is getting narrow.

In the above "Accuracy Curve" graph we can see that the blue marker indicates training accuracy and the orange marker indicates validation accuracy. After each epoch, we can see that the training accuracy is increasing as well as the validation accuracy is also increasing and the gap between training accuracy and validation accuracy is getting much narrow.

- **Unbalanced Dataset**

Table 5.10: Performance metrics of DistilBERT (Unbalanced data)

Label	Precision	Recall	F1-score	Support
Not Bully	0.86	0.84	0.85	1484
Religious	0.92	0.89	0.91	730
Sexual	0.82	0.88	0.84	867
Threat	0.69	0.75	0.72	160
Troll	0.77	0.76	0.77	983

Our train data for DistilBERT is 4247 (Unbalanced data). From the above classification report, we can see that the religious label is working best among all other labels. Its support is 730/4247 and the precision, recall, and f1-score are the best among all. The not bully label and sexual label are performing quite similarly. Which has also a very decent amount of good precision, recall, and f1-score. At last threat and troll label is also working very decent.

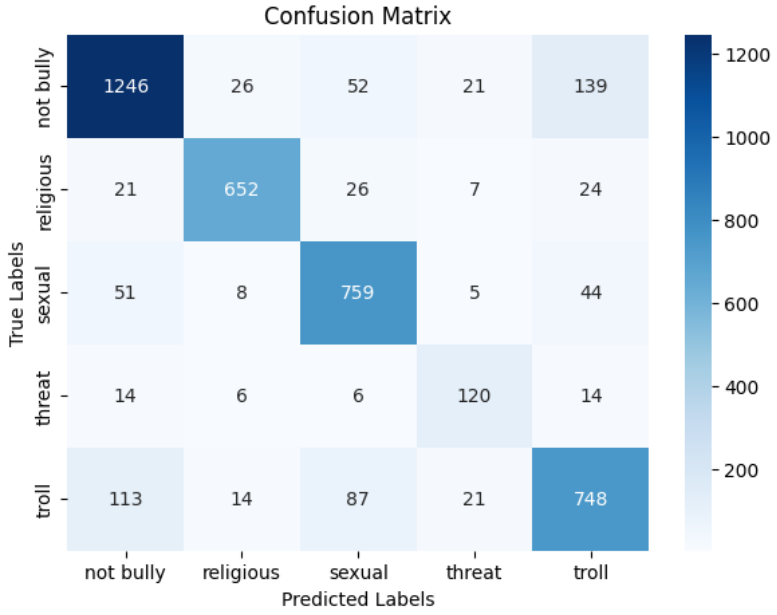

Figure 5.11: Heatmap of DistilBERT (Unbalanced data)

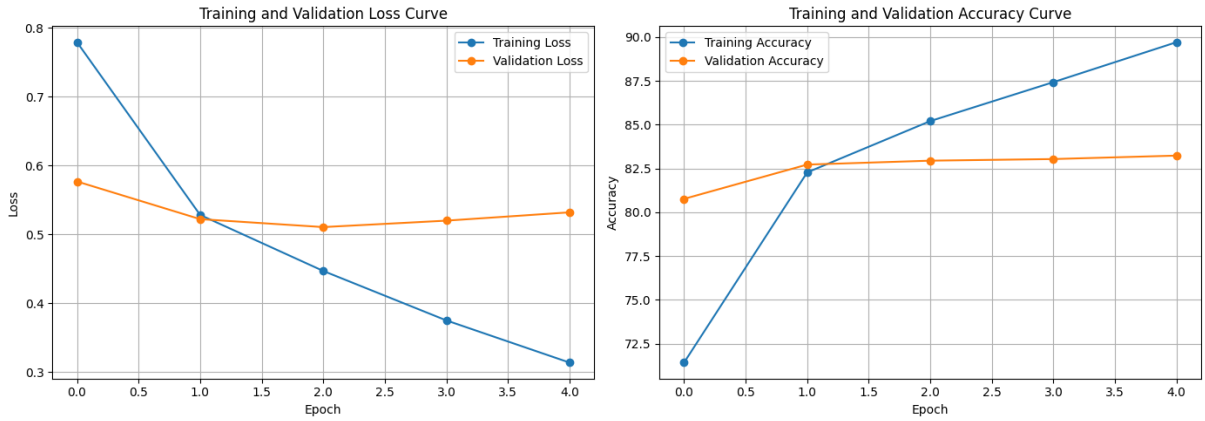

Figure 5.12: Loss and Accuracy Curve of DistilBERT (Unbalanced data)

From the above "Loss Curve" graph we can see that after each epoch, the training loss is decreasing as well as the validation loss is also decreasing but in the end validation loss increased slightly. It's because the model is biased towards the majority data label.

In the above "Accuracy Curve" graph we can see after each epoch, the training accuracy is increasing as well as the validation accuracy is also increasing slightly. However, the gap is wide because of imbalanced data.

Table 5.11: Performance of DistilBERT (Balance VS Unbalance data)

Dataset	**Accuracy**	**Precision**	**Recall**	**F1-score**
Balanced	0.9087	0.9085	0.9087	0.9079
Unbalanced	0.8345	0.8358	0.8345	0.8348

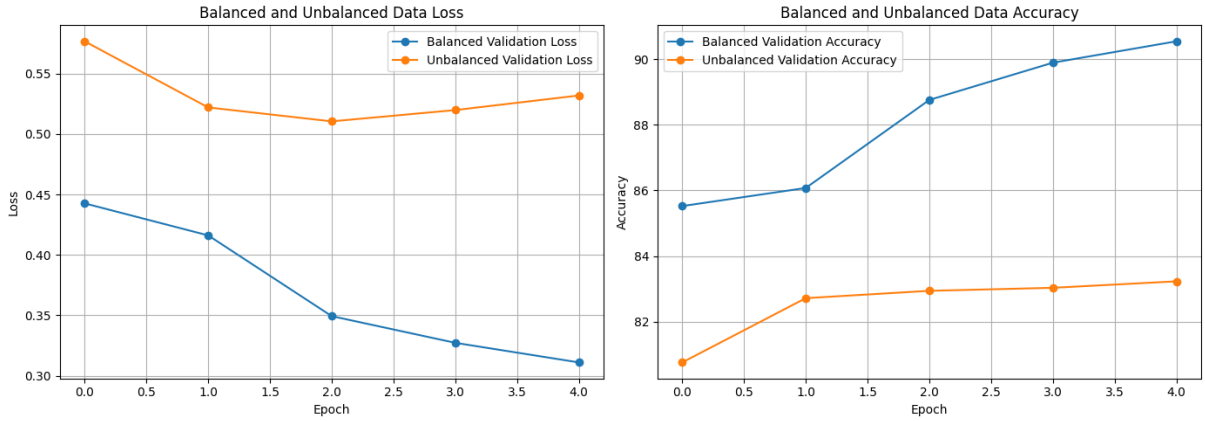

Figure 5.13: Loss and Accuracy of DistilBERT (Balance VS Unbalance)

The balanced dataset is performing well from all. Both loss and accuracy are good from others.

5.2.2 XLM-RoBERTa Base

We implemented XLMRoberta Base for our text classification task. "XLMRobertaTokenizer" was used and then loaded with the "xlm-roberta-base" pre-trained model. It takes the help of a BERT-based model for sequence classification and specifically uses the “xlm-roberta-base” pre-trained weights. Tokenizing has been done using the text with the XLMRobertaTokenizer.

- **Balanced Dataset**

Table 5.12: Performance metrics of XLM-RoBERTa Base (Balanced data)

Label	Precision	Recall	F1-score	Support
Not Bully	0.84	0.89	0.86	1450
Religious	0.98	0.95	0.97	1452
Sexual	0.93	0.91	0.92	1494
Threat	0.98	1.00	0.99	1459
Troll	0.88	0.84	0.86	1409

The above classification report shows that the threat label is working best among all other labels. Then comes religious and sexual. Which also has a very decent amount of good precision, recall, and f1-score but the label religious has higher scores than sexual. At last, Not bully and troll label is also working very decent.

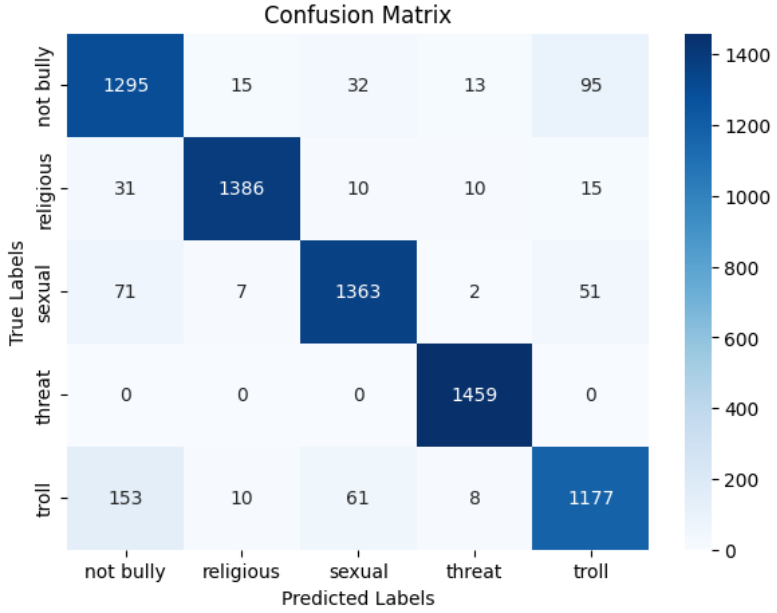

Figure 5.14: Heatmap of XLM-RoBERTa Base (Balanced data)

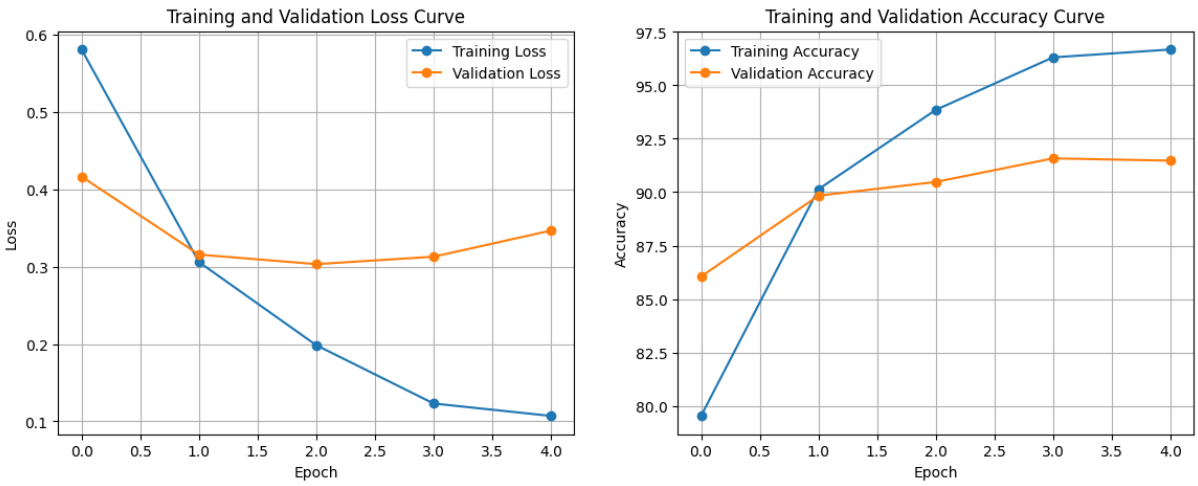

Figure 5.15: Curve of XLM-RoBERTa Base (Balanced data)

The curve indicates that there is a slight increase in validation loss, but the model is performing quite well and the gap between training loss and validation loss is quite narrow.

- **Unbalanced Dataset**

Table 5.13: Performance metrics of XLM-RoBERTa (Unbalanced data)

Label	**Precision**	**Recall**	**F1-score**	**Support**
Not Bully	0.86	0.83	0.85	1484
Religious	0.93	0.89	0.91	730
Sexual	0.81	0.87	0.84	867
Threat	0.65	0.78	0.71	160
Troll	0.77	0.76	0.76	983

According to the above table, the not bully and sexual labels perform similarly, but not bully performs slightly better than sexual. Religious is doing the best. Performance scores of the religious label are higher among all. Not bully label consist of more data but the model is performing well for religious label.

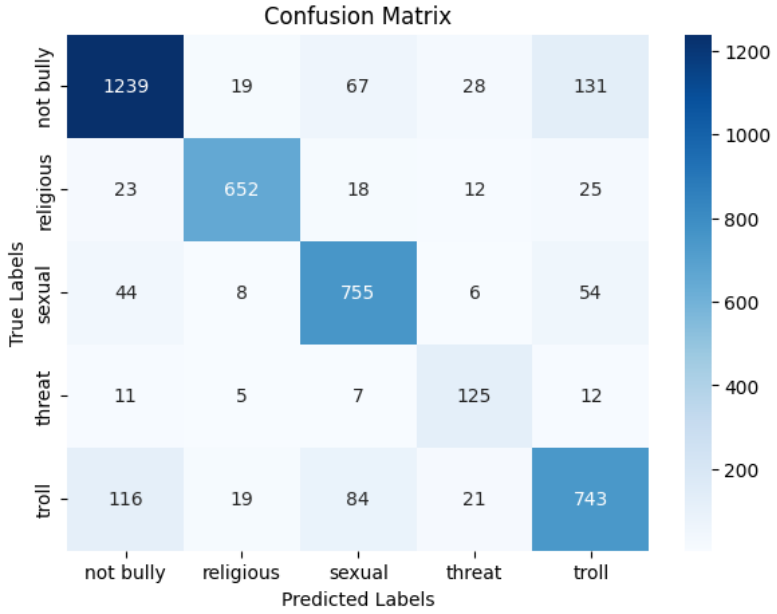

Figure 5.16: Heatmap of XLM-RoBERTa (Unbalanced data)

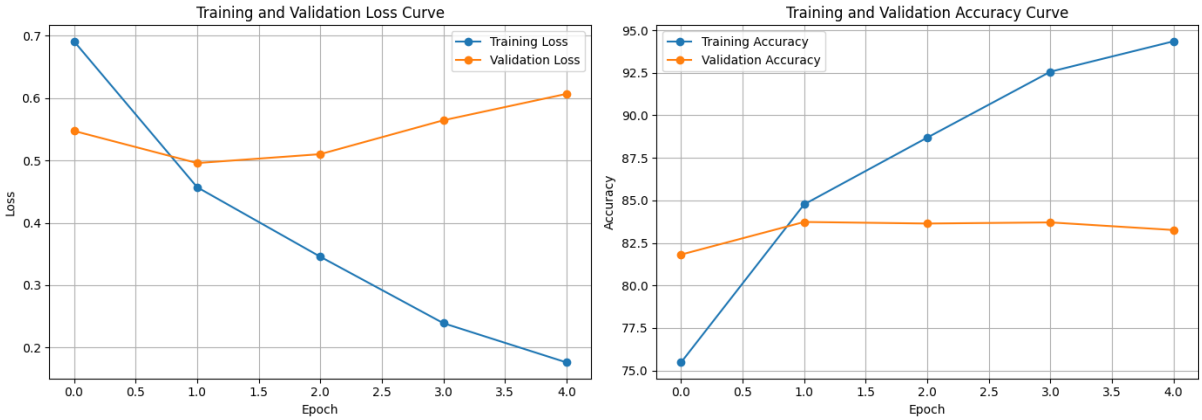

Figure 5.17: Curve of XLM-RoBERTa (Unbalanced data)

From the 3rd epoch, the loss is increasing. As a result, the model is more relying on training data and it will not work well for unseen data. The model is more inclined towards data having the label of "not bully", which causes a bias in the system.

Table 5.14: Performance of XLM-RoBERTa (Balance VS Unbalance data)

Dataset	Accuracy	Precision	Recall	F1-score
Balanced	0.9196	0.9204	0.9196	0.9196
Unbalanced	0.8319	0.8343	0.8319	0.8325

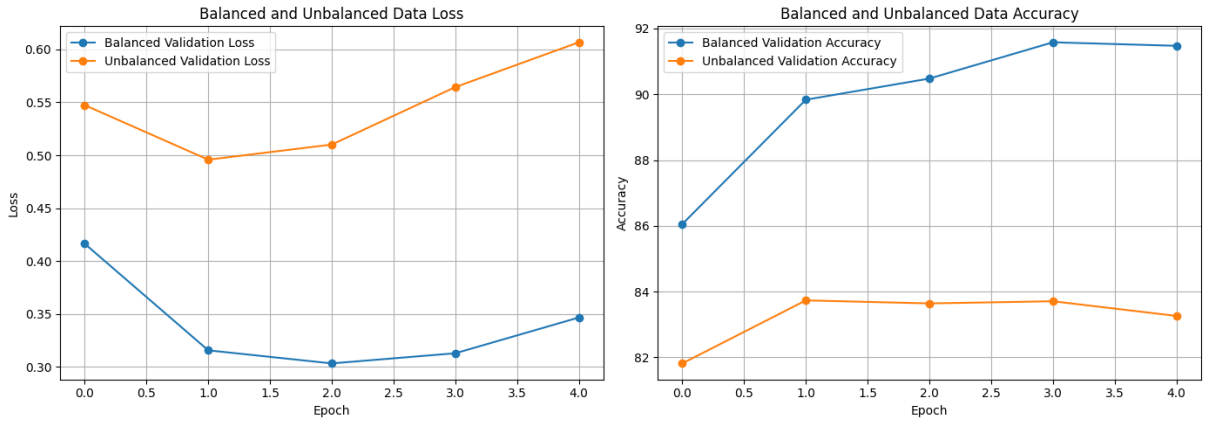

Figure 5.18: Loss and Accuracy Curve of XLM-RoBERTa Base (Balanced VS Unbalanced)

The balanced dataset is performing well in every case. Both loss and accuracy are performing well for the balanced dataset.

5.2.3 Bangla BERT

For our comment classification task, we have implemented BanglaBERT. "BertTokenizer" was used from “sagorsarker/bangla-bert-base” pretrained model. We used early-stopping criteria here as well.

- **Balanced Dataset**

Table 5.15: Performance metrics of Bangla BERT (Balanced data)

Label	Precision	Recall	F1-score	Support
Not Bully	0.86	0.85	0.86	1450
Religious	0.95	0.96	0.96	1494
Sexual	0.89	0.93	0.91	1494
Threat	0.98	0.99	0.99	1459
Troll	0.87	0.81	0.84	1409

The threat label is the most effective. Then there’s religious and sexual, and finally, not bully and troll.

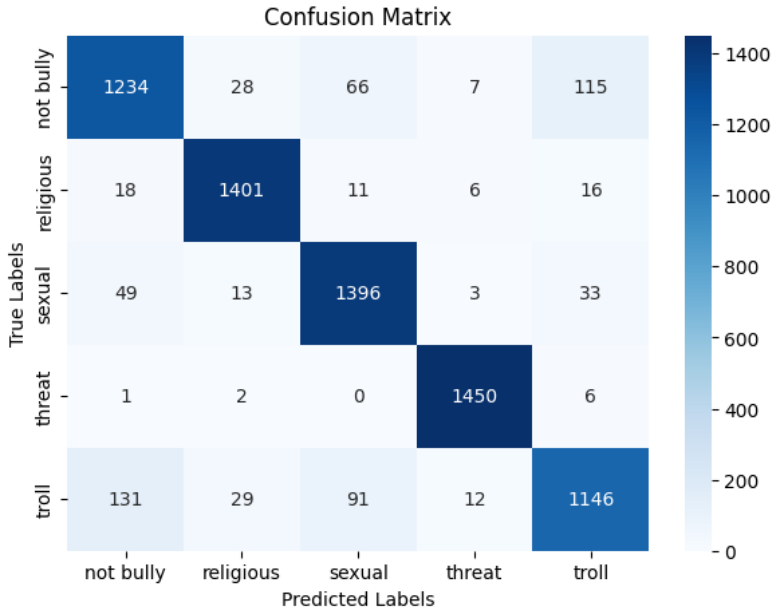

Figure 5.19: Heatmap of BanglaBERT (Balanced data)

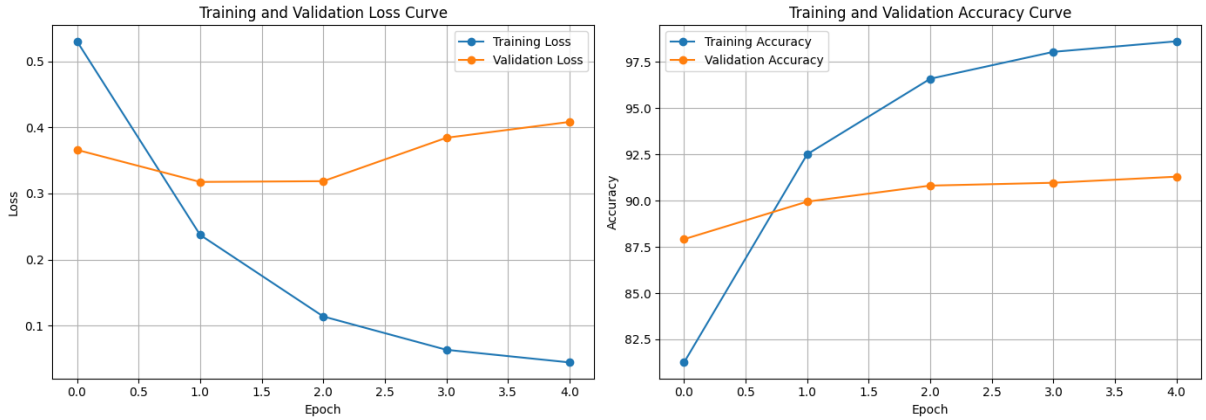

Figure 5.20: Curve of BanglaBERT (Balanced data)

The graph shows that after the 3rd epoch, validation loss increased and remained the same from the 4th to the 5th epoch.

In the above “Accuracy Curve” graph we can see that the gap between training accuracy and validation accuracy is getting much narrower.

- **Unbalanced Dataset**

Table 5.16: Performance metrics of Bangla BERT (UnBalanced data)

Label	Precision	Recall	F1-score	Support
Not Bully	0.84	0.83	0.83	1484
Religious	0.88	0.91	0.89	730
Sexual	0.83	0.81	0.82	867
Threat	0.76	0.76	0.76	160
Troll	0.75	0.75	0.75	983

From the above classification report, we can see that the overall performance of religious label and not bully label is good among all other labels. The precision, recall, and f1-score are quite good and balanced among all. Then comes sexual which has also a good performance. Lastly, the threat and troll label is also working well.

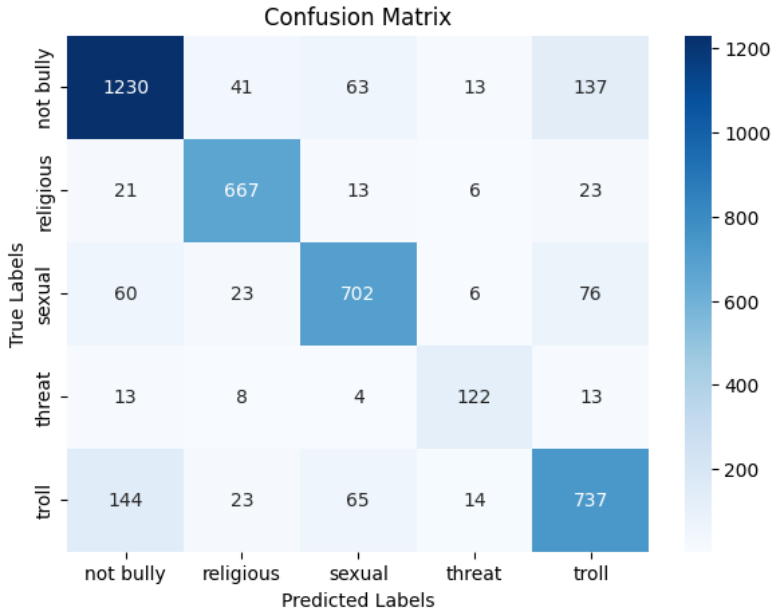

Figure 5.21: Heatmap of BanglaBERT (Unbalanced data)

In the figure 5.22, the "Loss Curve" graph shows that both losses are decreasing in 1st epoch. But from the 2nd epoch, the validation loss increased and still increased through the 5th epoch.

The "Accuracy Curve" graph illustrates that after the second epoch, validation accuracy drops slightly and continues to drop until the fourth epoch. Then, after the fourth epoch, increases.

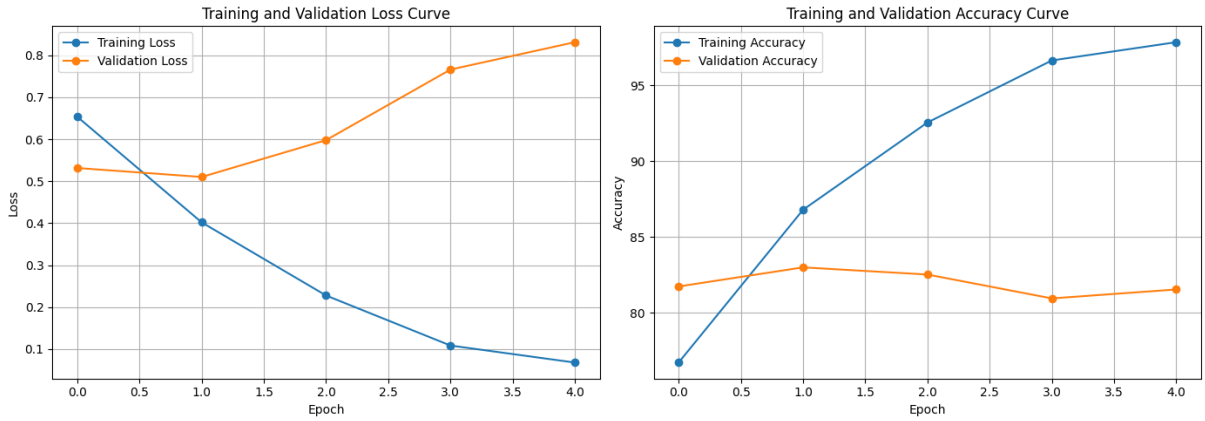

Figure 5.22: Curve of BanglaBERT (Unbalanced data)

Table 5.17: Performance of BanglaBERT (Balance VS Unbalance data)

Dataset	Accuracy	Precision	Recall	F1-score
Balanced	0.9123	0.9116	0.9123	0.9116
Unbalanced	0.8187	0.8184	0.8187	0.8184

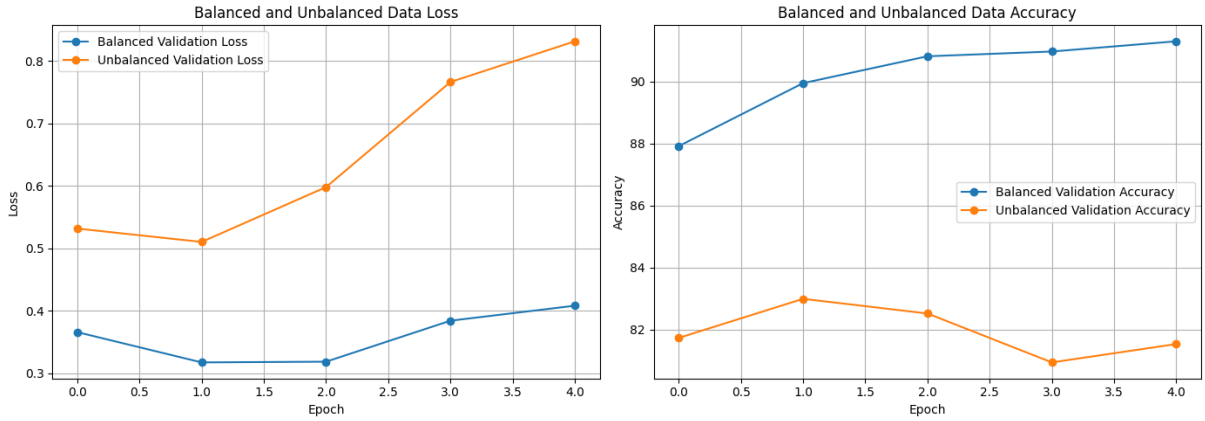

Figure 5.23: Loss and Accuracy Curve of BanglaBERT (Balanced VS Unbalanced data)

Instead of imbalanced data, balanced data performs well. As a result, the model is not biased on the label of the majority of the data.

5.2.4 Combination of CNN and Bi-LSTM

The first layer in this model was the embedding layer, which used categorical data to construct continuous vector representations. It was fine-tuned during training based on the number of unique vocabulary, the dimension of the word embeddings, the length of the input sequences, and the pre-trained word embeddings. Subsequently, 50% of the input units were arbitrarily set to zero using the SpatialDropout1D layer to avoid overfitting. We

then employed a pooling layer called MaxPooling1D with a two-unit window, which reduces the spatial dimensions of the data while maintaining important features. Two 1D convolutional layers were added following MaxPooling1D and SpatialDropout1D, having 128 filters with a kernel size of 5 and 64 filters with a kernel size of 3, respectively. ReLU was employed as the activation function in each scenario. After that, the output is sent to the 64-unit Bidirectional LSTM (BiLSTM) layer, which is an essential component of the architecture. Bidirectional BiLSTM allows the model to capture dependencies in both forward and reverse directions, which improves the model's comprehension of long-range patterns. Methods of regularization were also used to reduce overfitting. GlobalAveragePooling1D helped to average the data across the sequence to produce a fixed-length representation of the data. Then, we introduced a fully connected dense layer with 64 units and ReLU activation, followed by a dropout layer that sets 30% of the preceding layer's units to zero during training to prevent overfitting. Finally, a final dense layer of five units with a softmax activation function—representing the output classes—was applied to the model.

We have applied the same model architecture and techniques to both unbalanced and balanced data to check how the model performs in both cases. The results for both unbalanced and balanced data are given below:

- **Balanced Dataset**

Table 5.18: Performance metrics of CNN+BiLSTM (Balanced data)

Label	**Precision**	**Recall**	**F1-score**	**Support**
Not Bully	0.80	0.81	0.81	1477
Religious	0.93	0.91	0.92	1508
Sexual	0.81	0.88	0.84	1529
Threat	0.95	0.98	0.96	1577
Troll	0.84	0.75	0.79	1517

From the above classification report, we can see that the threat label performs the best out of all the labels. Its precision, recall, and f1-score are the highest of all, and its support is 1577. Next, we can observe that the troll label performs the worst in terms of recall and f1-score, and not bully in terms of precision.

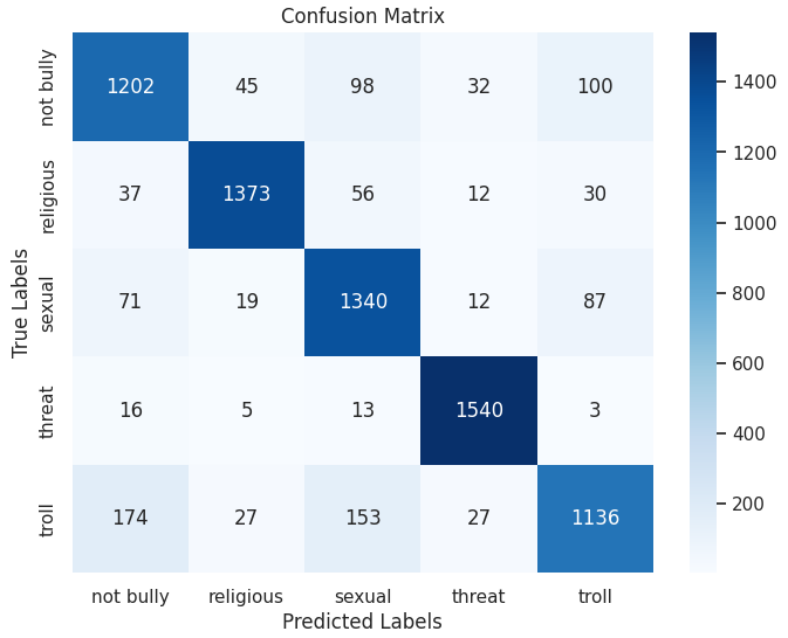

Figure 5.24: Heatmap of CNN+BiLSTM (Balanced data)

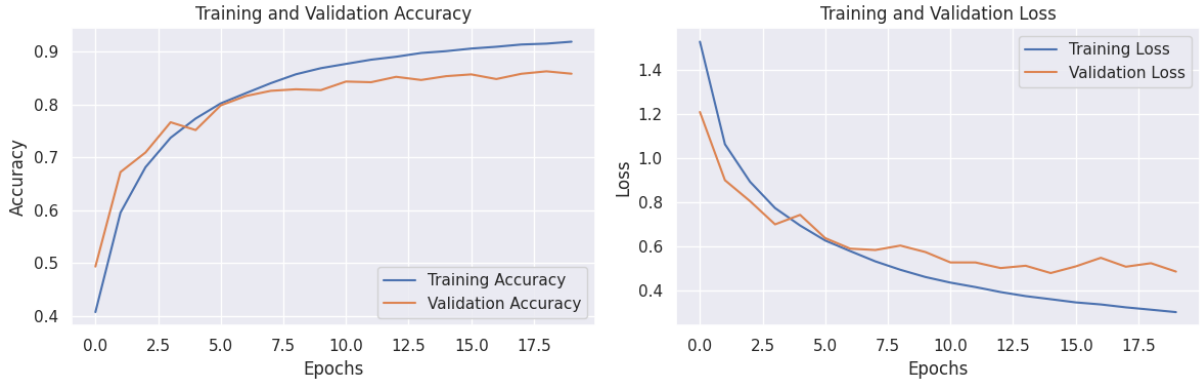

Figure 5.25: Loss and Accuracy Curve of CNN+BiLSTM (Balanced data)

From the above "Loss Curve" graph, we can see that with each epoch, both the validation and training losses are declining. And the gap between training loss and validation loss is getting narrower.

From the "Accuracy Curve" graph, we can observe that both the training and validation accuracy increase after each epoch from start to end, and the gap between training accuracy and validation accuracy is getting much narrower. So, we can say that both training and validation accuracy and loss of balanced data are performing better.

The model is stopped early after 20 epochs because if it is trained for more than 20 epochs, it might show signs of overfitting.

- **Unbalanced Dataset**

Table 5.19: Performance metrics of CNN+BiLSTM (Unbalanced data)

Label	Precision	Recall	F1-score	Support
Not Bully	0.77	0.86	0.81	1523
Religious	0.91	0.85	0.85	761
Sexual	0.82	0.75	0.78	876
Threat	0.80	0.62	0.70	152
Troll	0.69	0.72	0.70	1048

From the above classification report, we can see that, the religious label performs the best out of all the labels. Its precision and f1-score are the highest of all, and its support is 761. Next, we can observe that the threat label performs the worst in terms of recall and troll in terms of precision. The threat and troll labels perform the same in terms of f1-score, both having the lowest score.

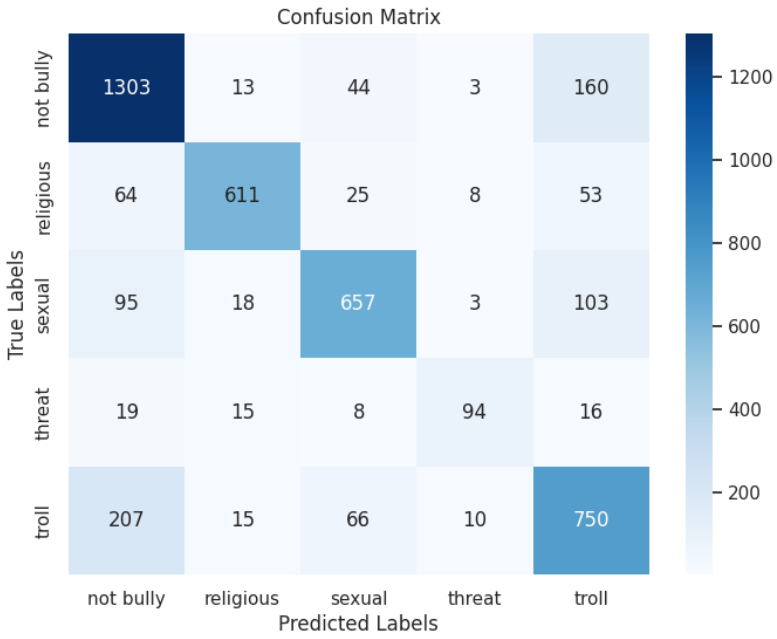

Figure 5.26: Heatmap of CNN+BiLSTM (Unbalanced data)

From the below figure 5.27, we can see that in the “Loss Curve” both the validation and training losses are declining with each epoch. Ultimately, the training loss continues to decline, while the validation loss fluctuates slightly.

From the “Accuracy Curve” graph, we can observe that both the training and validation accuracy increase after each epoch. After that, it keeps increasing, maintaining a certain level.

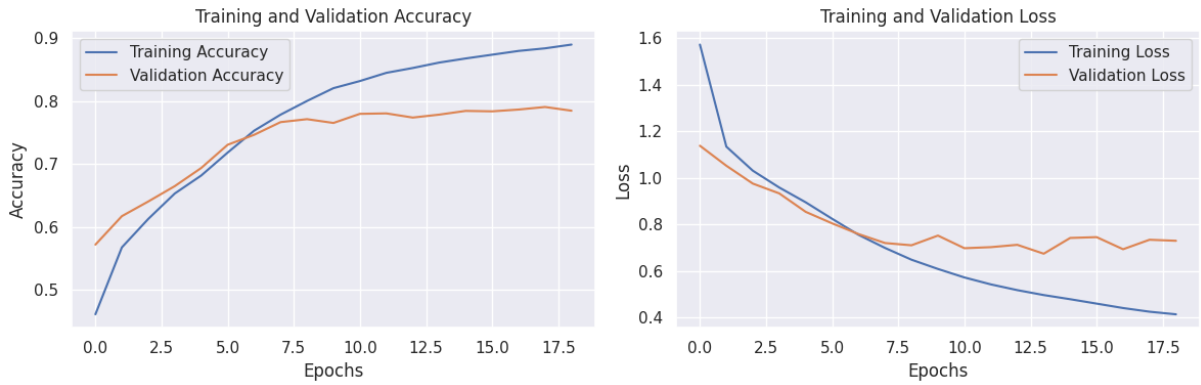

Figure 5.27: Loss and Accuracy Curve of CNN+BiLSTM (Unbalanced data)

So, we can see that both training and validation accuracy show improvement over time, which can be a positive sign. Also, the gap between training and validation accuracy and loss is not that large, indicating there might be no overfitting. The model is stopped early after 19 epochs because if it is trained for more than 19 epochs, it might show signs of overfitting.

Table 5.20: Performance of CNN+BiLSTM (Balance VS Unbalance data)

Dataset	**Accuracy**	**Precision**	**Recall**	**F1-score**
Balanced	0.8663	0.8668	0.8663	0.8657
Unbalanced	0.7833	0.7877	0.7833	0.7834

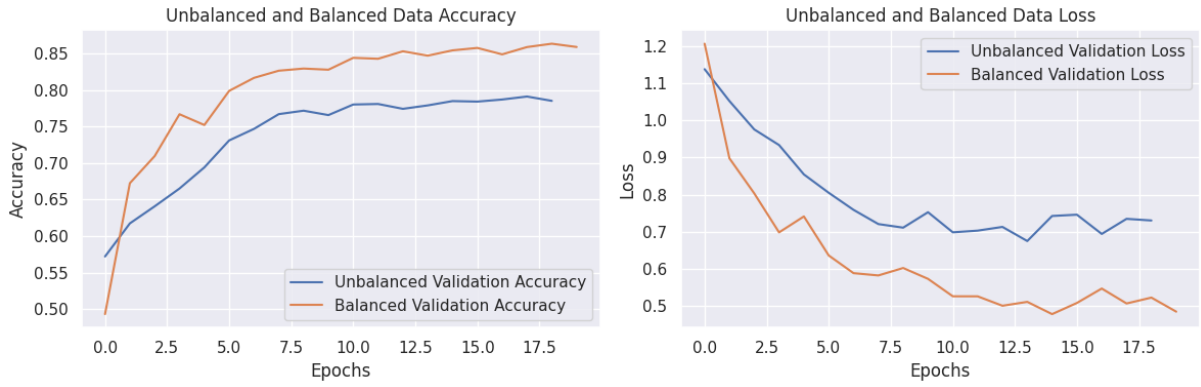

Figure 5.28: Loss and Accuracy of CNN+BiLSTM (Balance VS Unbalance)

It is evident from the above figure that the validation loss and accuracy are both operating satisfactorily for the balanced dataset. When compared to an unbalanced dataset, the balanced dataset's validation accuracy increases more and more with time. Also, the validation loss of the balanced dataset is consistently lower compared to the unbalanced dataset.

5.2.5 Combination of CNN and LSTM

We implemented a combination of CNN and LSTM for our text classification task. All the layers were added sequentially one after another. The first layer was the embedding layer, which in this instance also had parameters similar to the previously described model. A 1D convolutional layer with 64 filters, a kernel size of 5, and ReLU as the activation function was added after the embedding layer. The spatial dimensions of the input data are then reduced by adding a 1D average pooling layer, which takes the average over a window of data. It helps in controlling overfitting. The output is then sent to the 100-unit LSTM layer, which has tanh as the activation function and a dropout of 0.3. Long-range dependencies in sequences can be captured by LSTM. Two fully connected dense layers one with 128 units and the other with 64 units having ReLU activation were then added after and before LSTM and dropout layers respectively. Finally, a dense layer of five units having a softmax activation function was added.

The combination of CNN and LSTM model architecture and techniques were applied to both unbalanced and balanced data to check how the model performs in both cases. The results for both unbalanced and balanced data are given below:

- **Balanced Dataset**

Table 5.21: Performance metrics of CNN+LSTM (Balanced data)

Label	**Precision**	**Recall**	**F1-score**	**Support**
Not Bully	0.72	0.88	0.79	1477
Religious	0.94	0.91	0.92	1508
Sexual	0.89	0.83	0.86	1529
Threat	0.97	0.95	0.96	1577
Troll	0.82	0.75	0.79	1517

From the above classification report, we can see that the threat label performs the best out of all the labels. Its precision, recall, and f1-score are the highest of all, and its support is 1577. Next, we can see that in terms of precision and f1-score, the not bully label performs the poorest. Additionally, the troll has the lowest recall and f1-score scores. Furthermore, not bully and troll labels have the same f1-score value.

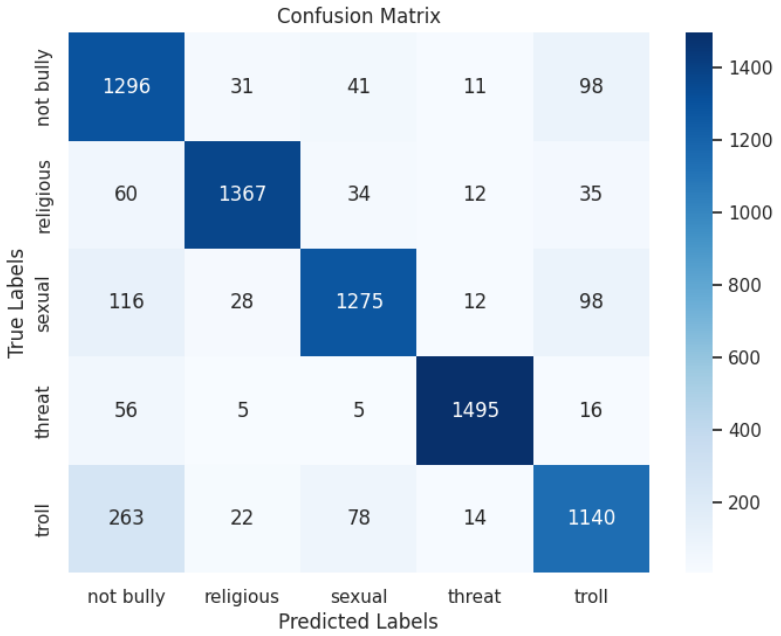

Figure 5.29: Heatmap of CNN+LSTM (Balanced data)

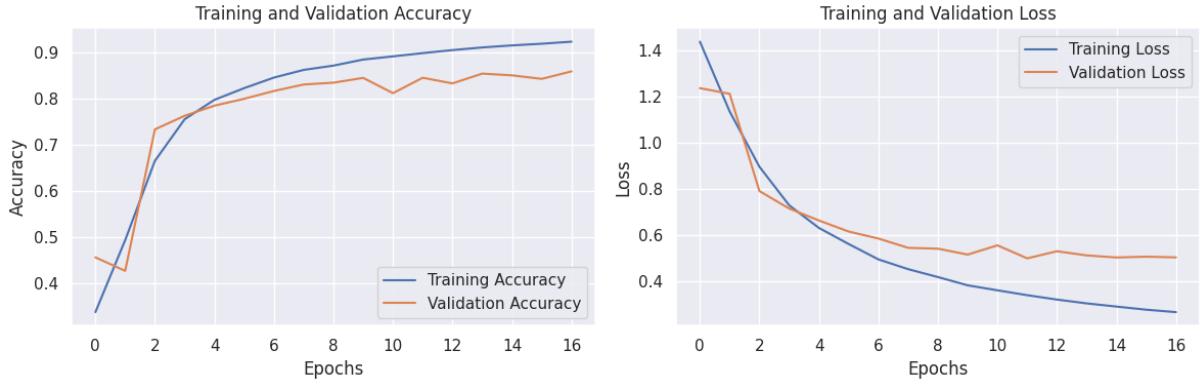

Figure 5.30: Loss and Accuracy Curve of CNN+LSTM (Balanced data)

From the above "Loss Curve" graph, we can see that with each epoch, both the validation and training losses are declining.

From the "Accuracy Curve" graph, we can observe that both the training and validation accuracy increase after each epoch.

Therefore, we may conclude that both the difference between training and validation losses and the difference between training and validation accuracy decreases with each epoch. Thus, we may observe improved performance in terms of training and validation accuracy as well as loss of balanced data. In the end, the model is stopped early after 17 epochs because if it is trained for more than 17 epochs, it might show signs of overfitting.

- **Unbalanced Dataset**

Table 5.22: Performance metrics of CNN+LSTM (Unbalanced data)

Label	Precision	Recall	F1-score	Support
Not Bully	0.77	0.86	0.81	1523
Religious	0.87	0.82	0.85	761
Sexual	0.73	0.76	0.75	876
Threat	0.76	0.49	0.59	152
Troll	0.73	0.64	0.68	1048

From the above classification report, we can see that the religious label performs the best out of all the labels. Its precision and f1-score are the highest of all, and its support is 761. Not bully label has the highest recall in our case. Next, we can observe that the threat label performs the worst in terms of recall and f1-sore, while both the sexual and troll labels perform the worst in the case of precision having the same values.

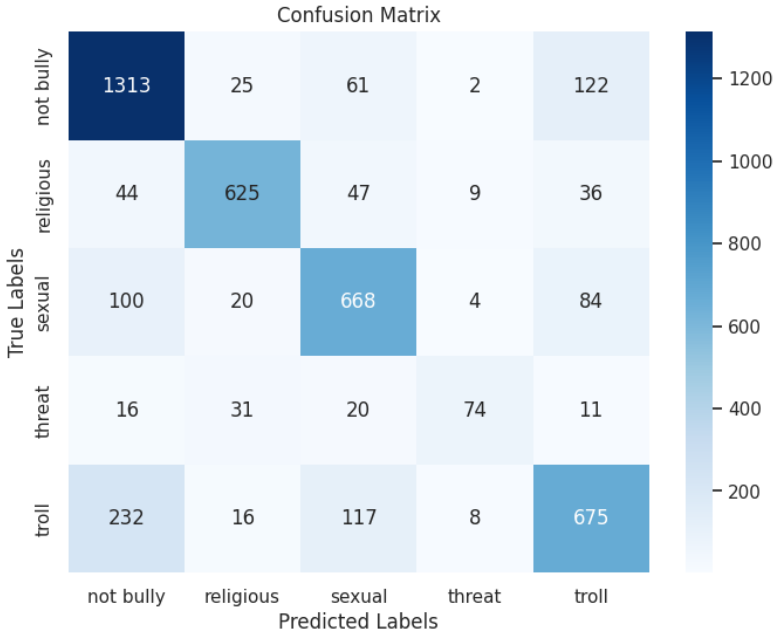

Figure 5.31: Heatmap of CNN+LSTM (Unbalanced data)

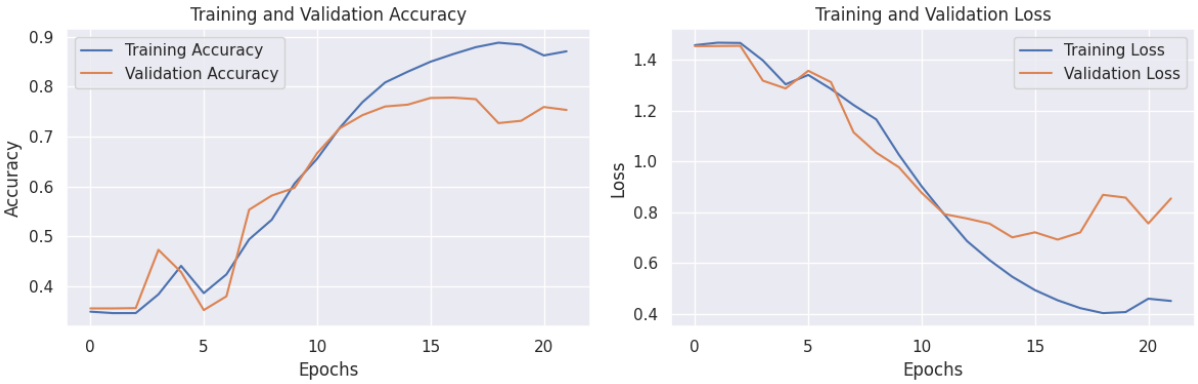

Figure 5.32: Loss and Accuracy Curve of CNN+LSTM (Unbalanced data)

From the above "Loss Curve" graph, we can observe that both the validation and training losses are declining with each epoch. However, in the case of a validation loss, we can see fluctuations. We can infer from the report generated during training and the graph that the validation loss continues to rise slightly after 15 epochs until somewhat decreases after some epochs. Also, the gap between training loss and validation loss is somewhat higher.

From the "Accuracy Curve" graph, we can observe that after each epoch, the accuracy for both training and validation increases. Although, at some epochs, the validation accuracy faces a fall but it still keeps increasing, and we can see fluctuations. Around epoch 15, we can see the validation accuracy seems to maintain a certain level, and there is little improvement afterward.

Though the model is stopped early after 22 epochs, it does not show any improvement in accuracy after epoch 17. If the model is trained for more than 22 epochs, it might overfit.

Table 5.23: Performance of CNN+LSTM (Balance VS Unbalance data)

Dataset	**Accuracy**	**Precision**	**Recall**	**F1-score**
Balanced	0.8640	0.8704	0.8640	0.8651
Unbalanced	0.7695	0.7696	0.7695	0.7668

It is evident from the figure in 5.33 that, the validation loss and accuracy are both operating satisfactorily for the balanced dataset. When compared to an unbalanced dataset, the balanced dataset's validation accuracy increases with lesser fluctuations. Also, the validation loss of the balanced dataset shows signs of a constant decrease compared to the unbalanced dataset.

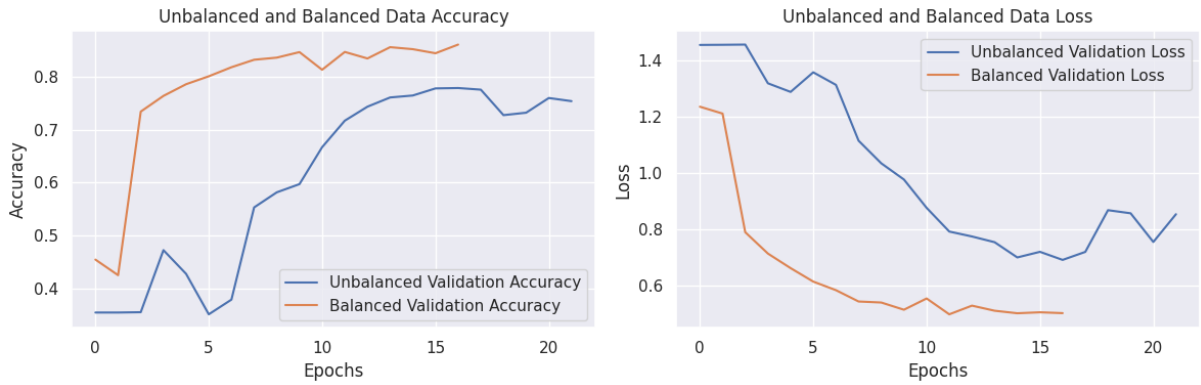

Figure 5.33: Loss and Accuracy of CNN+LSTM (Balance VS Unbalance)

5.2.6 Bi-LSTM

We implemented the Bi-LSTM model for our text classification task. All the layers were added sequentially one after another. The embedding layer was the initial layer, and in this case, its parameters were the same as the models that were previously mentioned. After that, 20% of the input units were arbitrarily set to zero in a SpatialDropout1D layer. The output was then sent to the 64-unit Bi-LSTM layer, which has tanh as the activation function and a dropout of 0.3. Next, a dense layer with 128 units that had l2 regularizer and ReLU activation was added. Next, a batch normalizing layer is applied, which helps to enhance the training stability. Finally, a dense layer of five units having a softmax activation function was added.

- **Balanced Dataset**

Table 5.24: Performance metrics of Bi-LSTM (Balanced data)

Label	Precision	Recall	F1-score	Support
Not Bully	0.78	0.82	0.80	1477
Religious	0.94	0.93	0.94	1508
Sexual	0.84	0.87	0.85	1529
Threat	0.95	0.97	0.96	1577
Troll	0.82	0.75	0.78	1517

From the above classification report, we can see that the threat label performs the best out of all the labels. Its precision, recall, and f1-score are the highest of all, and its support is 1577. After that, the precision, recall, and f1-score of religious labels come in second place. Next, we can observe that the label not bully has the lowest precision and the troll label yields the lowest recall and f1-score.

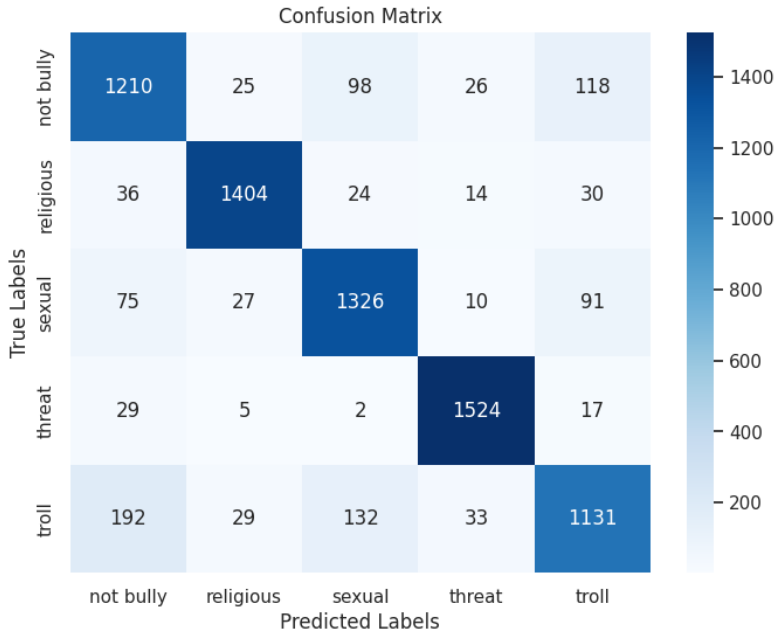

Figure 5.34: Heatmap of BiLSTM (Balanced data)

From the below "Loss Curve" graph, we can see that with each epoch, both the validation and training losses are declining. And the gap between training loss and validation loss is getting narrower.

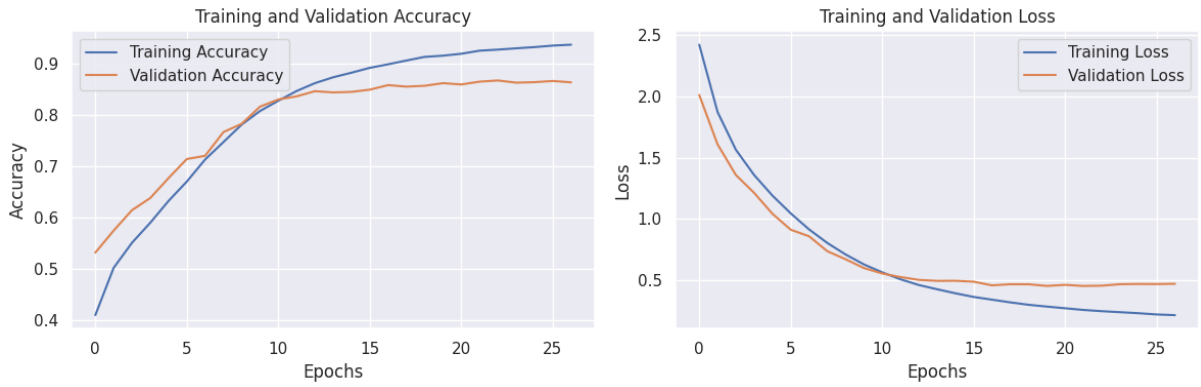

Figure 5.35: Loss and Accuracy Curve of BiLSTM (Balanced data)

From the "Accuracy Curve" graph, we can observe that both the training and validation accuracy increase after each epoch from start to end. The gap between training and validation accuracy is getting narrower, indicating a good balance. Furthermore, the model's validation accuracy and loss show little progress after epoch 23, they have no significant improvement. The model is stopped early after 27 epochs because if it is trained for more than 27 epochs, it might overfit.

- **Unbalanced Dataset**

Table 5.25: Performance metrics of BiLSTM (Unbalanced data)

Label	Precision	Recall	F1-score	Support
Not Bully	0.76	0.87	0.81	1523
Religious	0.86	0.82	0.84	761
Sexual	0.70	0.74	0.72	876
Threat	0.79	0.53	0.63	152
Troll	0.71	0.60	0.65	1048

From the above classification report, we can see that the religious label performs the best out of all the labels. Its precision and, f1-score are the highest of all, and its support is 761. Next, we can see that in terms of precision, the sexual label performs the poorest, and the troll label performs marginally better than it. When it comes to recall and f1-score, the threat label performs poorly. And when considering the f1-score, the troll label performs slightly better than the threat label.

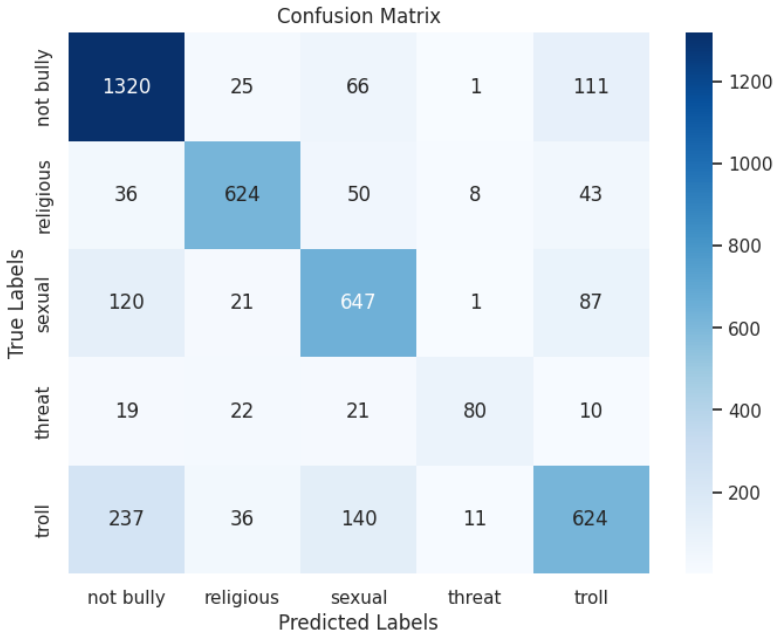

Figure 5.36: Heatmap of BiLSTM (Unbalanced data)

From the below "Loss Curve" graph of figure 5.37, we can see that with each epoch, both the validation and training losses are declining. And the gap between training loss and validation loss is getting narrower.

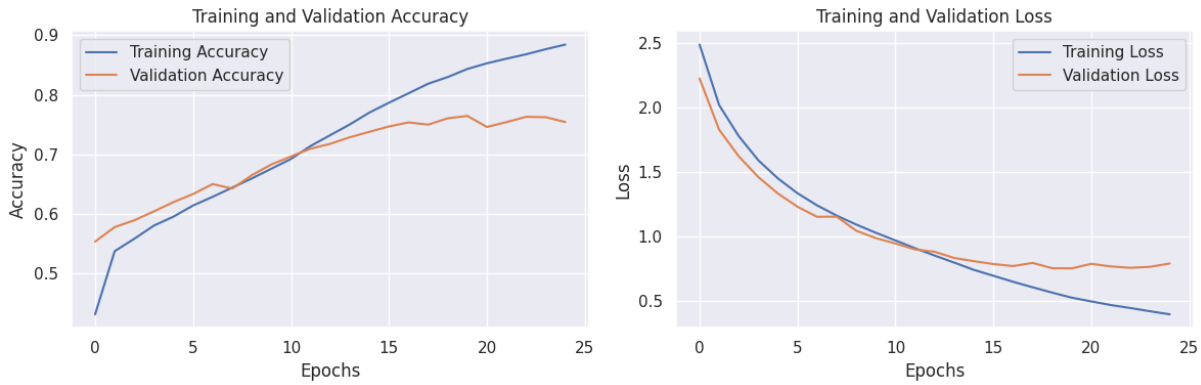

Figure 5.37: Loss and Accuracy Curve of BiLSTM (Unbalanced data)

From the "Accuracy Curve" graph, we can observe that both the training and validation accuracy increase after each epoch from start to end, and the accuracy curves are relatively close and there is no drastic shift between them. Furthermore, the model's validation accuracy and loss show little progress after epoch 20, instead, they fluctuate slightly. The model is stopped early after 25 epochs because if it is trained for more than 25 epochs, it might overfit.

Table 5.26: Performance of BiLSTM (Balance VS Unbalance data)

Dataset	**Accuracy**	**Precision**	**Recall**	**F1-score**
Balanced	0.8669	0.8667	0.8669	0.8663
Unbalanced	0.7557	0.7555	0.7557	0.7521

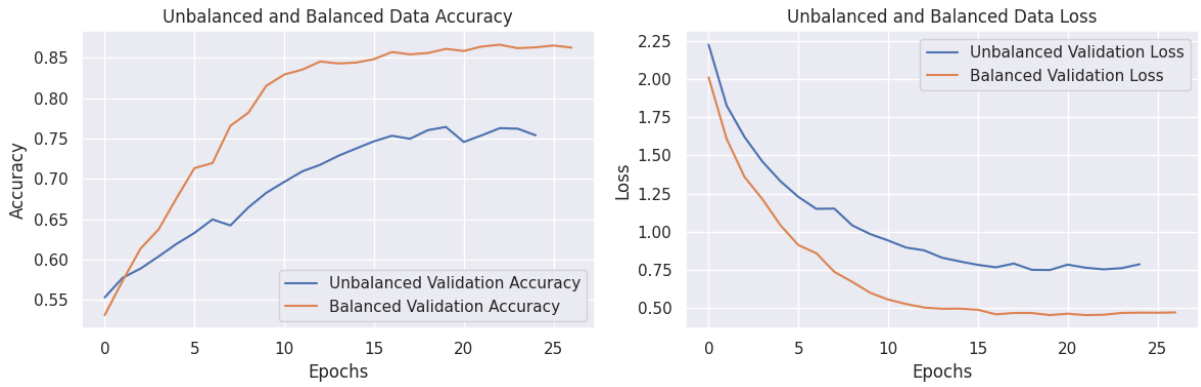

Figure 5.38: Loss and Accuracy of BiLSTM (Balance VS Unbalance)

It is evident from the above two figures that the validation loss and accuracy are both operating satisfactorily for the balanced dataset. The validation accuracy of the balanced dataset is significantly higher than that of the unbalanced dataset. In addition, compared to the unbalanced dataset, the balanced dataset's validation loss shows signs of a constant decrease.

5.2.7 LSTM

We implemented LSTM model for our text classification task. All the layers were added sequentially one after another. The embedding layer was the initial layer, and in this case, its parameters were the same as the models that were previously mentioned. After that, 20% of the input units were arbitrarily set to zero in a SpatialDropout1D layer. The output was then sent to the 100-unit LSTM layer, which has tanh as the activation function and a dropout of 0.2. Finally, a dense layer of five units having a softmax activation function was added.

- **Balanced Dataset**

Table 5.27: Performance metrics of LSTM (Balanced data)

Label	Precision	Recall	F1-score	Support
Not Bully	0.81	0.79	0.80	1477
Religious	0.95	0.92	0.94	1508
Sexual	0.85	0.86	0.86	1529
Threat	0.97	0.97	0.97	1577
Troll	0.78	0.81	0.79	1517

From the above classification report, we can see that the threat label performs the best out of all the labels. Its precision, recall, and f1-score are the highest of all, and its support is 1577. After that, the precision, recall, and f1-score of religious labels come in second place. Next, we can observe that the label troll has the lowest precision and f1-score, and the not bully label yields the lowest recall.

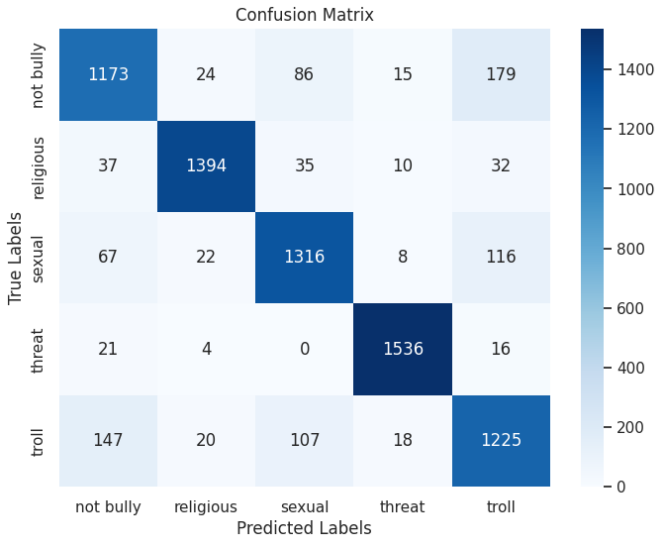

Figure 5.39: Heatmap of LSTM (Balanced data)

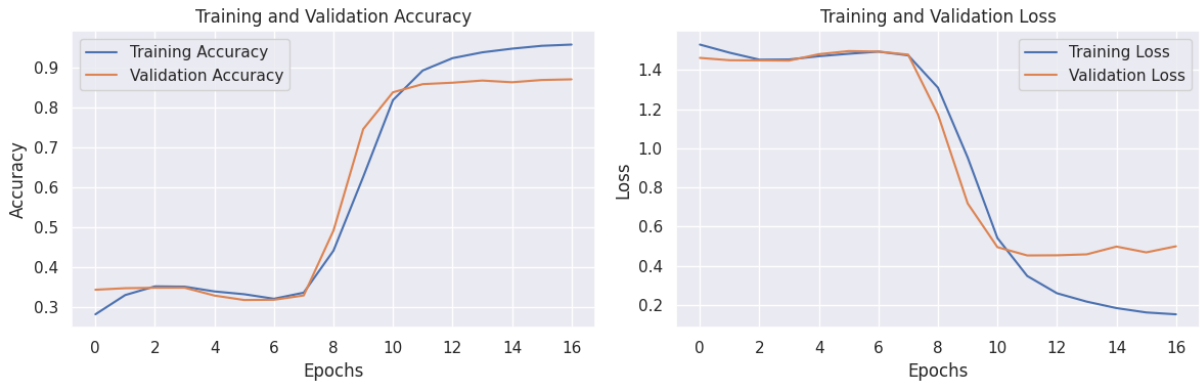

Figure 5.40: Loss and Accuracy Curve of LSTM (Balanced data)

From the above "Loss Curve" graph, we can see that with each epoch, both the validation and training losses are declining, and the validation loss values are becoming closer to the training loss. Additionally, the gap between training loss and validation loss gets smaller in the end than the unbalanced curve.

From the "Accuracy Curve" graph, we can observe that both the training and validation accuracy increase after each epoch from start to end. The gap between training and validation accuracy is getting narrower at the end, indicating a good balance. Furthermore, the model's validation accuracy and loss show little progress after epoch 14, they have no significant improvement. The model is stopped early after 17 epochs because if it is trained for more than 17 epochs, it might overfit.

- **Unbalanced Dataset**

Table 5.28: Performance metrics of LSTM (Unbalanced data)

Label	**Precision**	**Recall**	**F1-score**	**Support**
Not Bully	0.75	0.83	0.79	1523
Religious	0.91	0.79	0.84	761
Sexual	0.65	0.70	0.68	876
Threat	0.46	0.55	0.50	152
Troll	0.63	0.53	0.58	1048

From the above classification report, we can see that the religious label performs the best out of all the labels. Its precision and, f1-score are the highest of all, and its support is 761. Next, we can see that in terms of precision and f1-score, the threat label performs the poorest. And when considering the recall, the threat label performs slightly better than the troll label.

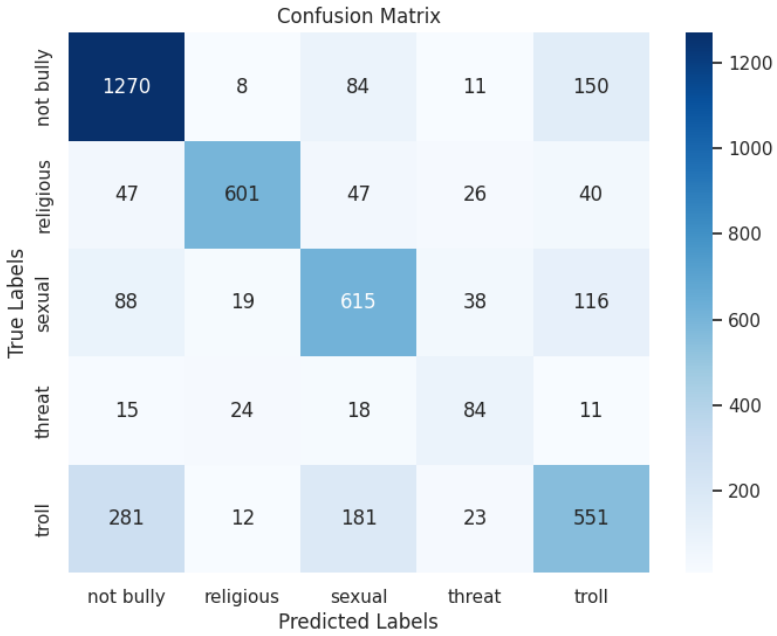

Figure 5.41: Heatmap of LSTM (Unbalanced data)

From the below "Loss Curve" graph, we can see that with each epoch, both the validation and training losses are declining which is a positive sign. After, epoch 9, the validation loss improves slightly. And, here the gap between training loss and validation loss is much more than any other model.

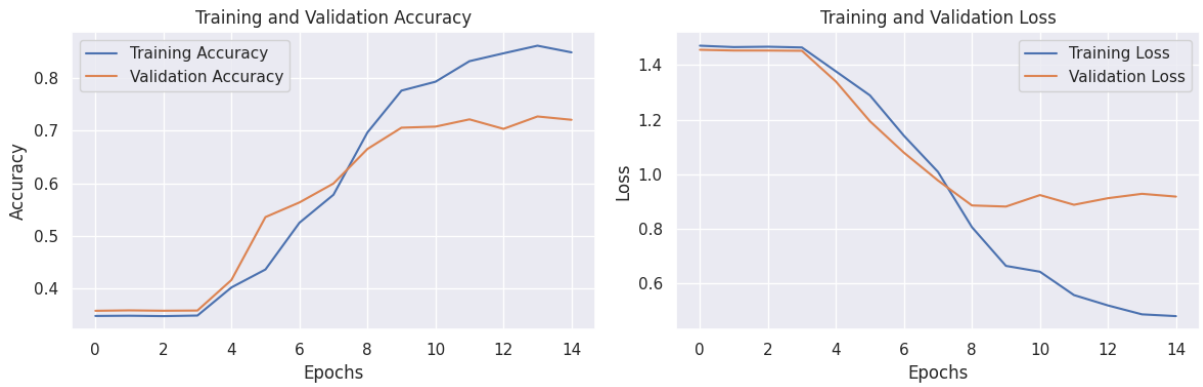

Figure 5.42: Loss and Accuracy Curve of LSTM (Unbalanced data)

From the "Accuracy Curve" graph, we can observe that both the training and validation accuracy increase after each epoch from start to end, and the difference between both the accuracy curves is not that much. The model is stopped early after 15 epochs because if it is trained for more than 15 epochs, it might overfit.

Table 5.29: Performance of LSTM (Balance VS Unbalance data)

Dataset	Accuracy	Precision	Recall	F1-score
Balanced	0.8773	0.8740	0.8733	0.8735
Unbalanced	0.7158	0.7182	0.7158	0.7139

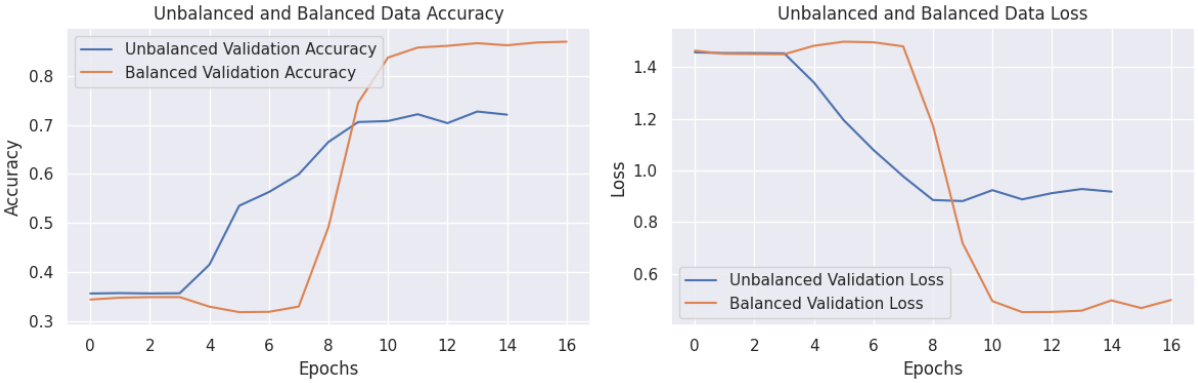

Figure 5.43: Loss and Accuracy of LSTM (Balance VS Unbalance)

It is evident from the above two figures that the validation loss and accuracy are both operating satisfactorily for the balanced dataset. The validation accuracy of the balanced dataset is significantly higher than that of the unbalanced dataset. In addition, compared to the unbalanced dataset, the balanced dataset's validation loss shows signs of a constant decrease.

5.2.8 m-BERT

We employed m-BERT using the "BertForSequenceClassification" class from the Hugging Face Transformers library, using the pre-trained "bert-base-multilingual-cased" model. In this task, we worked with binary classification, where 10,000 data from our English dataset translated into Bangla and added to our Bangla Dataset which contained around 44,001 data. The preprocessing involves encoding sentences using the BertTokenizer, ensuring that special tokens are added, and handling padding and truncation to a specified maximum length. Additionally, attention masks are created to distinguish between actual tokens and padding tokens in the encoded sequences. For optimization, the AdamW optimizer is employed with a learning rate of 2e-5 and a batch size of 16. These hyperparameters are commonly used in fine-tuning BERT models. The scheduler adjusts the learning rate during training, starting with a warm-up phase where the learning rate gradually increases from 0 to the specified rate, followed by a linear decay over the training steps. The forward and backward propagation are performed, and gradients are clipped to prevent exploding gradients. The training loss is accumulated and normalized by the number of batches. After each epoch of training, the model is evaluated on a validation set.

- **Unbalanced Dataset**

Table 5.30: Performance metrics of M-BERT Base (Unbalanced data)

Label	Precision	Recall	F1-score	Support
Not Bully	0.83	0.82	0.82	4605
Bully	0.93	0.93	0.93	11587

The model achieved an overall accuracy of 90%.The model performs well in distinguishing between the two classes, with particularly high precision, recall, and F1-score for the "Bully" class. For the "Not Bully" class, the F1 score, which balances precision and recall, was calculated at 82%, suggesting a good trade-off between identifying Bully and Not Bully classes.

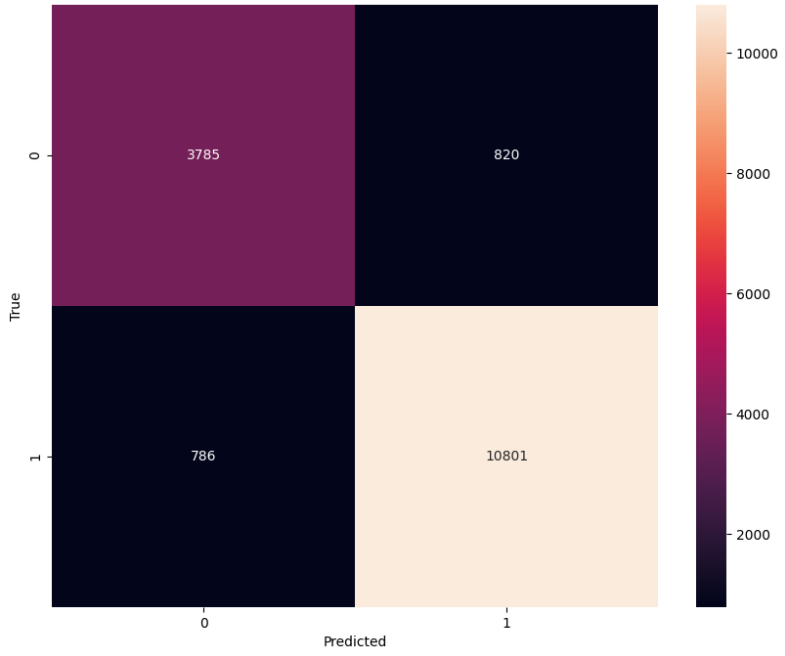

Figure 5.44: Heatmap of m-BERT (Unbalanced data)

The confusion matrix revealed that the model had 3785 true positive predictions, 820 false positives, 786 false negatives, and 10801 true negatives.

The training loss consistently decreases, indicating that the model is learning to fit the training data better and indicating convergence.The validation loss and accuracy show some variability, which could be attributed to the model's ability to generalize to unseen data. A slight increase in validation loss in later epochs might indicate potential overfitting.

Overall, the model appears to perform well on the training set, achieving high accuracy. However, monitoring the validation metrics is crucial to ensure that the model generalizes well to new, unseen data.

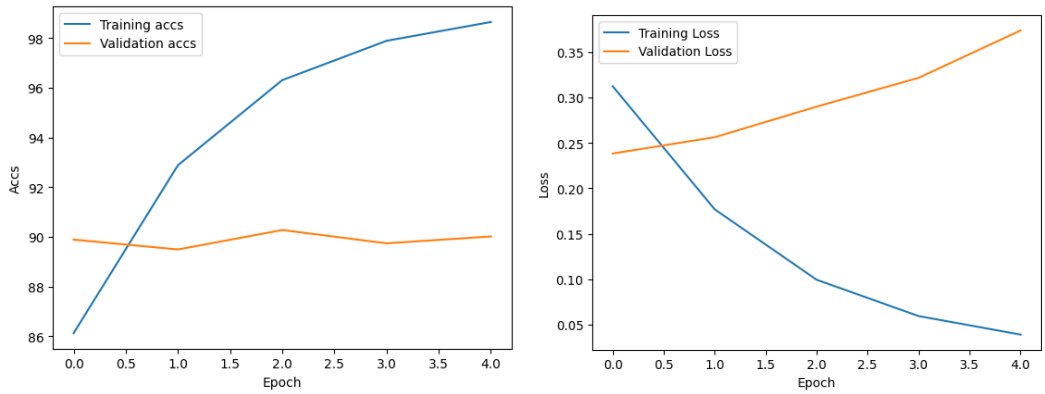

Figure 5.45: Loss and Accuracy of m-BERT (Unbalance)

- **Balanced Dataset**

Table 5.31: Performance metrics of M-BERT Base (Balanced data)

Label	Precision	Recall	F1-score	Support
Not Bully	0.93	0.91	0.92	11654
Bully	0.91	0.93	0.92	11530

In table 5.31, For the "Not Bully" class, the model has a precision of 0.93, indicating that 93% of instances predicted as "Not Bully" were correct. The recall is 0.91, meaning that 91% of all actual "Not Bully" instances were correctly identified. The F1-score, a balance between precision and recall, is 0.92.

For the "Bully" class, the model has a precision of 0.91, indicating that 91% of instances predicted as "Bully" were correct. The recall is 0.93, meaning that 93% of all actual "Bully" instances were correctly identified. The F1-score, a harmonic mean of precision and recall, is 0.92.

In summary, the model performs well in both classes, with high precision, recall, and F1-scores for both "Not Bully" and "Bully" classes. The F1-scores indicate a good balance between precision and recall, suggesting that the model effectively identifies instances of both classes while minimizing false positives and false negatives.

From the below Figure 5.46, we get 10630 instances correctly predicted as "Not Bully" class, 1024 instances incorrectly predicted as "Not Bully" class when they are actually "Bully", 783 instances incorrectly predicted as "Not Bully" when they are actually "Not Bully" and 10,747 instances correctly predicted as "Bully".

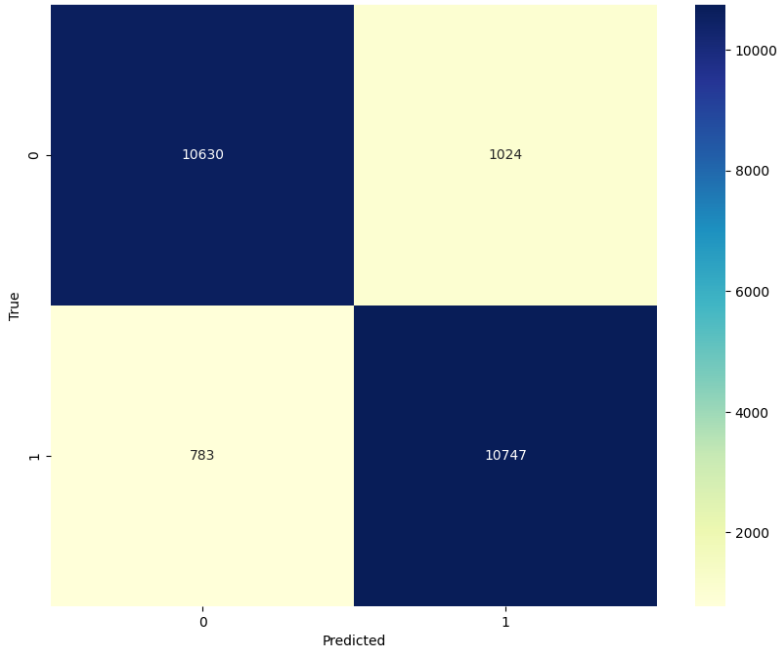

Figure 5.46: Heatmap of m-BERT (Balanced data)

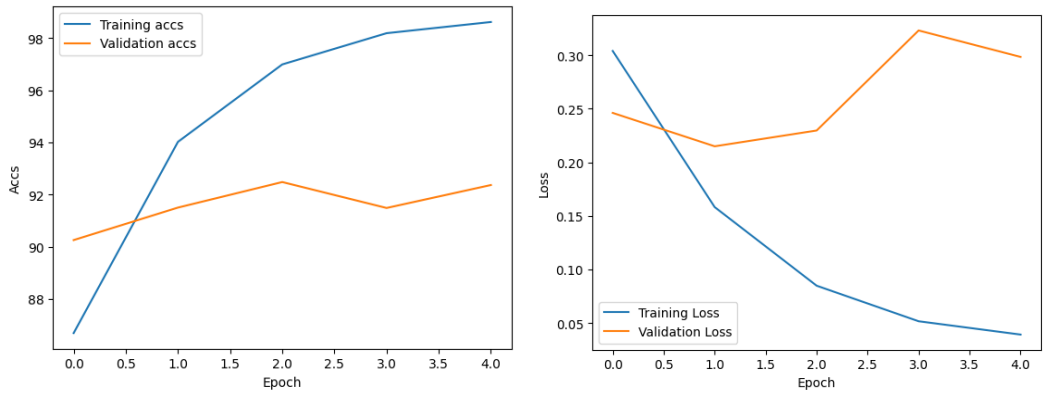

Figure 5.47: Loss and Accuracy of m-BERT (Balance)

In Figure 5.47, We can see that, the training accuracy steadily increases, suggesting that the model is becoming more proficient at classifying the training data. Validation accuracy shows improvement in the early epochs but may plateau or slightly decrease in later epochs, which could be a sign of overfitting.

The training loss consistently decreases over epochs, indicating that the model is learning and fitting the training data better with each epoch.The validation loss initially decreases but then starts to increase slightly after 3 epochs. This may indicate that the model is starting to overfit the training data. The model achieves high accuracy on both the training and validation sets, with the highest training accuracy reaching 98% in the final epoch. With test data, the model get accuracy around 92%.

In table 5.32, The balanced dataset has a higher accuracy compared to the unbalanced one, indicating better overall correctness. Precision measures the accuracy of the positive pre-

Table 5.32: Performance of m-BERT (Balance VS Unbalance data)

Dataset	Accuracy	Precision	Recall	F1-score
Balanced	0.92	0.92	0.92	0.92
Unbalanced	0.90	0.90	0.90	0.90

dictions. In this case, both the balanced and unbalanced datasets have the same precision, suggesting an equal rate of correct. Both the balanced and unbalanced datasets have the same recall comparing to their own Accuracy, Preciosion and F1-score.The F1-score is the harmonic mean of precision and recall. Both the balanced and unbalanced datasets have the same F1-score comparing to their own Accuracy, Precision and Recall.

In conclusion, the balanced dataset outperforms the unbalanced dataset in terms of accuracy.Precision, recall, and F1-score are the same for both balanced and unbalanced datasets in this comparison. The higher accuracy in the balanced dataset suggests better overall model performance, taking into account both true positives and true negatives.

5.2.9 IndicBERT

We employed an Indic-BERT pre-trained model from the Hugging Face Transformers library. The model was trained on a dataset of 53,998 labeled cyberbullying Bangla data, consisting of 43,998 samples from Kaggle and 10,000 samples translated from our English dataset. The dataset was preprocessed to remove some punctuations and tokenized for input into the model using the 'ai4bharat/indic-bert' Autotokenizer. The training process involved a batch size of 16, a learning rate of 2e-5, and underwent 4 epochs.We split the dataset into 70% of the training dataset, 15% of the testing dataset, and 15% of the validation dataset.

In the training process, the training data is reshuffled except for the first epoch. The model's state is saved if the validation accuracy improves. The performance of our sentiment analysis model was evaluated using precision, recall, and F1-score metrics.

- **Unbalanced Dataset**

Table 5.33: Performance metrics of IndicBERT (Unbalanced data)

Label	Precision	Recall	F1-score	Support
Not Bully	0.85	0.82	0.83	2552
Bully	0.92	0.93	0.92	5548

In table 5.33, for the "Bully" class, the model demonstrates high precision, indicating that 92% of instances predicted as "Bully" were correct. The recall is also high at 0.93, mean-

ing that 93% of all actual "Bully" instances were correctly identified. The F1-score 0.92, suggesting overall good performance in distinguishing the "Bully" class.

For the "Not Bully" class, the precision is 0.85, indicating that 85% of instances predicted as "Not Bully" were correct. The recall is 0.82, meaning that 82% of all actual "Not Bully" instances were correctly identified. The F1-score, a harmonic mean of precision and recall, is 0.83, suggesting a reasonable balance between precision and recall for the "Not Bully" class.

The model performs well in both classes, with particularly high recall for the "Bully" class and balanced precision and f1-score for the "Not Bully" class. The F1-scores for both classes are also relatively high, indicating a good overall trade-off between precision and recall.

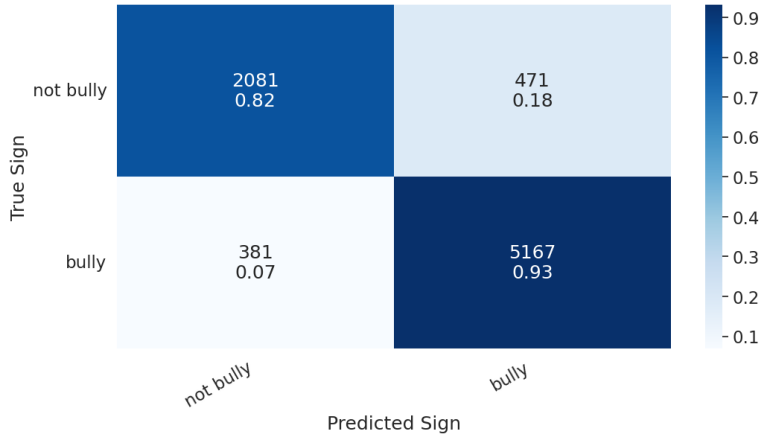

Figure 5.48: Heatmap of IndicBERT (Unbalanced data)

We got an accuracy of around 89%. The confusion matrix revealed that, 2081 data predicted as TP, 5167 data predicted as TN, 471 data predicted as FP, and 381 data predicted as FN.

Figure 5.49: Loss and Accuracy of IndicBERT (Unbalance)

In the Figure 5.49, The validation loss increases, suggesting that the model might be overfitting to the training data after a certain point. The gradual decrease in training loss indicates

that the model is improving its ability to fit the training data. Despite the increase in validation loss, the gradual increase in training accuracy indicates that the model is correctly predicting the target variable for a larger proportion of instances in the training sets, but the validation accuracy remains same may indicate that the model has learned as much as it can from the available data. .

- **Balanced Dataset**

Table 5.34: Performance metrics of IndicBERT (Balanced data)

Label	Precision	Recall	F1-score	Support
Not Bully	0.92	0.95	0.93	5548
Bully	0.94	0.91	0.93	5548

In table 5.34, we can see that, for the "not bully" class, the model has a precision of 0.92, indicating that 92% of instances predicted as "bully" were correct. The recall is 0.95, meaning that 95% of all actual "bully" instances were correctly identified. The F1-score, a balance between precision and recall, is 0.93.

For the "bully" class, the model has a precision of 0.94, indicating that 94% of instances predicted as "bully" were correct. The recall is 0.91, meaning that 91% of all actual "not bully" instances were correctly identified. The F1-score, a harmonic mean of precision and recall, is 0.93.

Both classes show balanced precision and recall, with high F1-scores, suggesting good overall performance in classifying instances into the "bully" and "not bully" categories. The support values indicate that there are 5548 instances for both classes in the dataset.

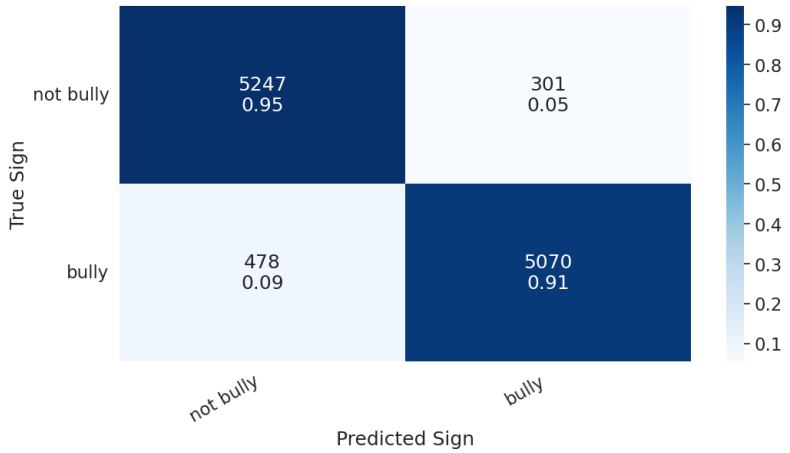

Figure 5.50: Heatmap of IndicBERT (Balanced data)

In Figure 5.50, we can see that, 5247 instances correctly predicted as "Not Bully" class, 301 instances incorrectly predicted as "Not Bully" class when they are actually belongs to "Bully"

class, 478 instances incorrectly predicted as "Bully" class when they are actually the "Not Bully" class and 5070 instances correctly predicted as "Bully" class.

Figure 5.51: Loss and Accuracy of IndicBERT (Balance)

In the Figure 5.51, After one epoch, the validation loss rises, indicating that the model may eventually start overfitting to the training set. The training loss is gradually declining, which suggests that the model is getting better at fitting the training set. The model is accurately predicting the target variable for a greater percentage of instances in both the training and validation sets, despite the progressive decrease in validation loss.

Loss and accuracy comparison of IndicBERT Large(Balance VS Unbalance):

Table 5.35: Performance of IndicBERT (Balance VS Unbalance data)

Dataset	Accuracy	Precision	Recall	F1-score
Balanced	0.93	0.93	0.93	0.93
Unbalanced	0.89	0.89	0.89	0.89

In table 5.35, The balanced dataset has a higher accuracy compared to the unbalanced one, indicating better overall correctness. The balanced dataset has a higher precision, suggesting a lower rate of false positives compared to the unbalanced dataset. Recall represents the ability of the model to capture all the positive instances. The balanced dataset has a higher recall than the unbalanced one. The balanced dataset shows a higher F1-score, indicating a better balance between precision and recall compared to the unbalanced dataset.

The balanced dataset outperforms the unbalanced dataset in terms of accuracy, precision, and F1-score. While the unbalanced dataset has a better trade-off between precision and recall, as indicated by the F1-score.

5.2.10 XLM RoBERTa large

We used a pre-trained XLM-RoBERTa model 'jplu/tf-xlm-roberta-large' from HuggingFace. We connect the embeddings to a series of layers: an LSTM layer with 128 units, batch normalization, two dense layers with 768 units each, ReLU activation, a dropout layer with a dropout rate of 0.1, and a final dense layer with 3 units and a sigmoid activation function. We compiled the model using sparse categorical cross-entropy as the loss function, the Adam optimizer, and accuracy as the evaluation metric.After that, we trained the model on the training data for 4 epochs with a batch size of 32. Validation data is used for monitoring performance during training. Callbacks, such as model checkpointing, early stopping, and learning rate reduction, are applied during training are essential for efficient model training and can help prevent overfitting, save the best model weights, stop training early if there is no improvement, and adjust the learning rate dynamically during training.

Table 5.36: Performance metrics of XLM Roberta large (Unbalanced data)

Label	**Precision**	**Recall**	**F1-score**	**Support**
Not Bully	0.80	0.66	0.72	2552
Bully	0.85	0.92	0.89	5549

In table 5.36, For the "Not Bully" class, the model has a precision of 0.80, indicating that 80% of instances predicted as "Not Bully" were correct. The recall is 0.66, meaning that 66% of all actual "Not Bully" instances were correctly identified. For the "Bully" class, the model has a precision of 0.85, indicating that 85% of instances predicted as "Bully" were correct. The recall is 0.92, meaning that 92% of all actual "Bully" instances were correctly identified.

In summary, the "Bully" class has higher precision, recall, and F1-score compared to the "Not Bully" class. The F1-scores for both classes indicate a good balance between precision and recall, providing a comprehensive view of the model's performance in the binary classification task.

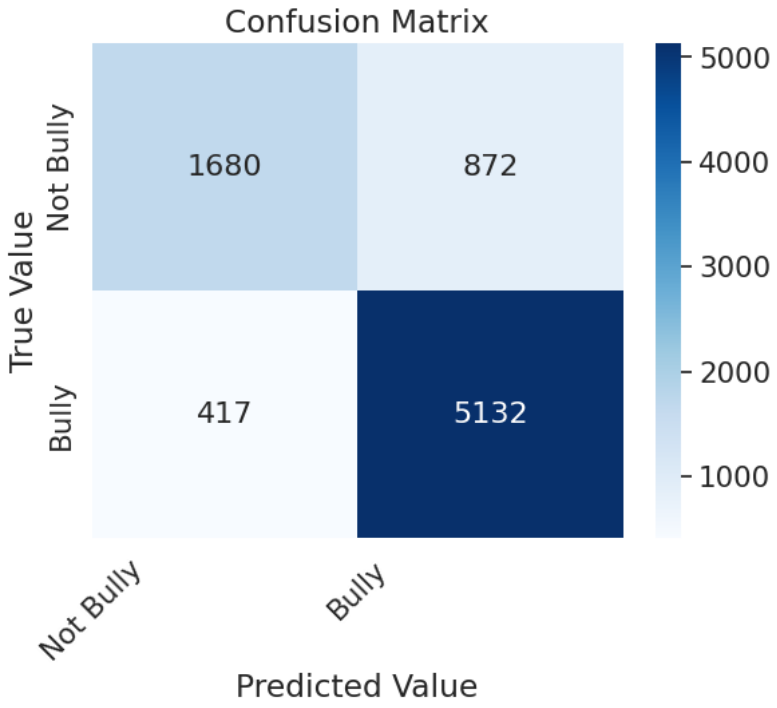

Figure 5.52: Heatmap of XLM Roberta large (Unbalanced data)

We got an accuracy of around 84%. The confusion matrix revealed that, 1680 data predicted as TP, 5132 data predicted as TN, 872 data predicted as FP, and 417 data predicted as FN.

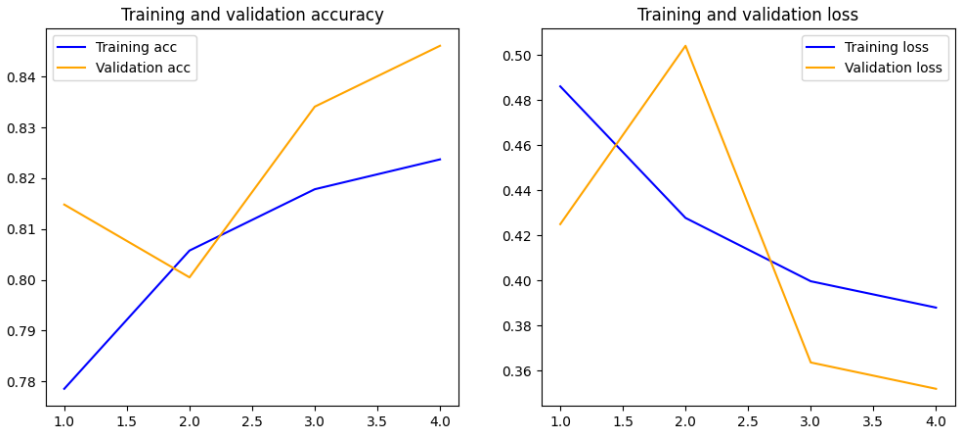

Figure 5.53: Loss and Accuracy of XLM RoBERTa large (Unbalance)

In Figure 5.53, we can see that, each of the epoch a decreasing in training and validation losses is visible, and accuracy increases, indicating improvement in the model's performance.The final model achieves a training accuracy of 85% and a validation accuracy of 84.60%, demonstrating good generalization performance.

- **Balanced Dataset**

Table 5.37: Performance metrics of XLM RoBERTa Large (Balanced data)

Label	Precision	Recall	F1-score	Support
Not Bully	0.80	0.91	0.85	5549
Bully	0.89	0.77	0.83	5548

In Table 5.37, we can see that, for the "Not Bully" class, the model has a precision of 0.80, indicating that 80% of instances predicted as "Not Bully" were correct. The recall is 0.91, meaning that 91% of all actual "Not Bully" instances were correctly identified. The F1-score, a balance between precision and recall, is 0.85.

For the "Bully" class, the model has a precision of 0.89, indicating that 89% of instances predicted as "Bully" were correct. The recall is 0.77, meaning that 77% of all actual "Bully" instances were correctly identified. The F1-score, a harmonic mean of precision and recall, is 0.83.

The model performs differently for the two classes, and the provided metrics offer insights that both classes little differences between precision and f1-score, suggesting good overall performance in classifying instances into the "bully" and "not bully" categories. Although there is an huge difference between recall and f1-score for "bully" class.

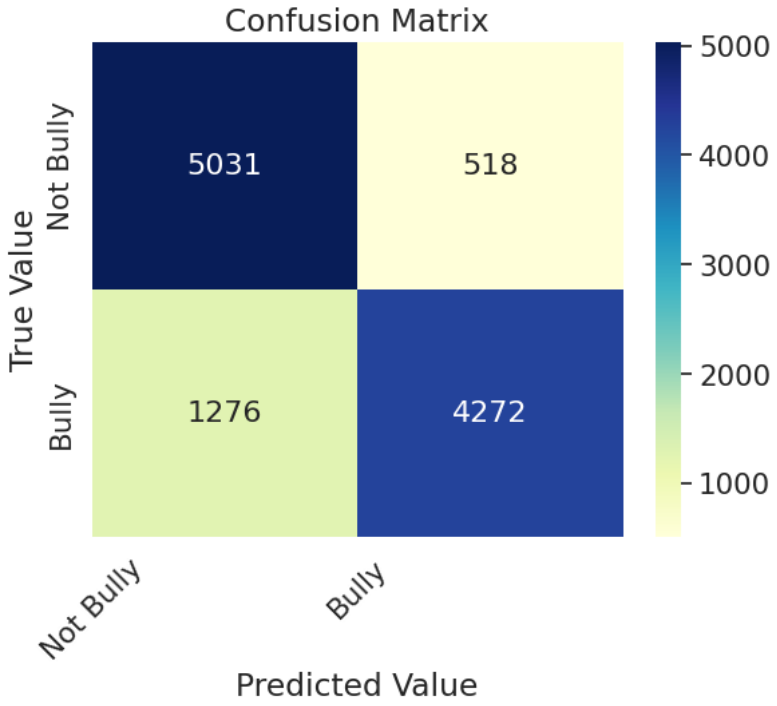

Figure 5.54: Heatmap of XLM RoBERTa (Balanced data)

In Figure 5.54, we can see that, 5031 instances correctly predicted as "Not Bully" class, 518 instances incorrectly predicted as "Bully" class when they are actually belongs to "Not Bully" class, 1276 instances incorrectly predicted as "Not Bully" class when they are actually the "Bully" class and 4272 instances correctly predicted as "Bully" class.

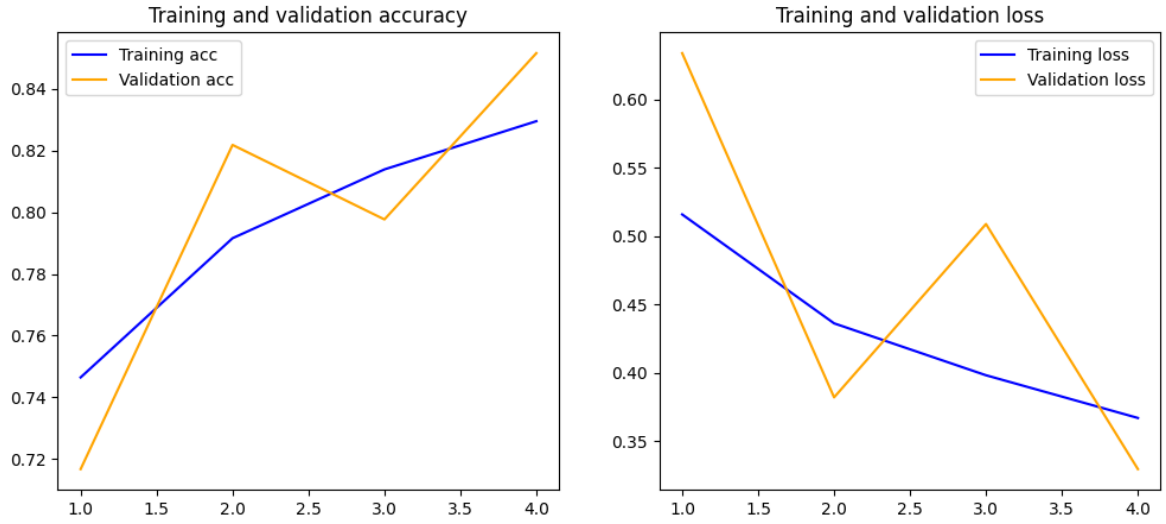

Figure 5.55: Loss and Accuracy of XLM RoBERTa Large (Balance)

In the Figure 5.55, We can see that, both the training and validation losses decrease over epochs, indicating improvement in the model's performance. Although, the validation loss increased after 2 epoch which indicates model's overfitting for a while, after 3 epoch the model again start decreasing which is an good sign of model's proper validation process. Training accuracy steadily increases, suggesting that the model is learning and fitting the training data better with each epoch.

Table 5.38: Performance of XLM RoBERTa Large (Balance VS Unbalance data)

Dataset	**Accuracy**	**Precision**	**Recall**	**F1-score**
Balanced	0.84	0.84	0.84	0.84
Unbalanced	0.84	0.84	0.84	0.84

In table 5.38, The accuracy for both the unbalanced dataset is equal to the balanced dataset.Precision measures the accuracy of the positive predictions. In this case, all the precision, recall and f1-score for both the balanced and unbalanced datasets are same.

So it is clearly seen that, there is not major differences even if we balanced the dataset. Both situation gives us the equal result.

5.3 Models Analysis

5.3.1 Machine Learning Models

English (Binary Class) : We have used four models to predict the accuracy. From the bar plot, we can see that the SVM is providing the highest accuracy which is about 95%. Multinomial Naive Bayes is performing lowest among all. It provides an accuracy of about 83%. The Logistic regression Model is performing slightly lower than both SVM and Random Forest but provides higher accuracy than Multinomial Naive Bayes. It's about 92%. At last, the Random Forest Classifier provides about 93% of accuracy. Figure 5.56 shows the binary classification plot for all models for the English dataset,

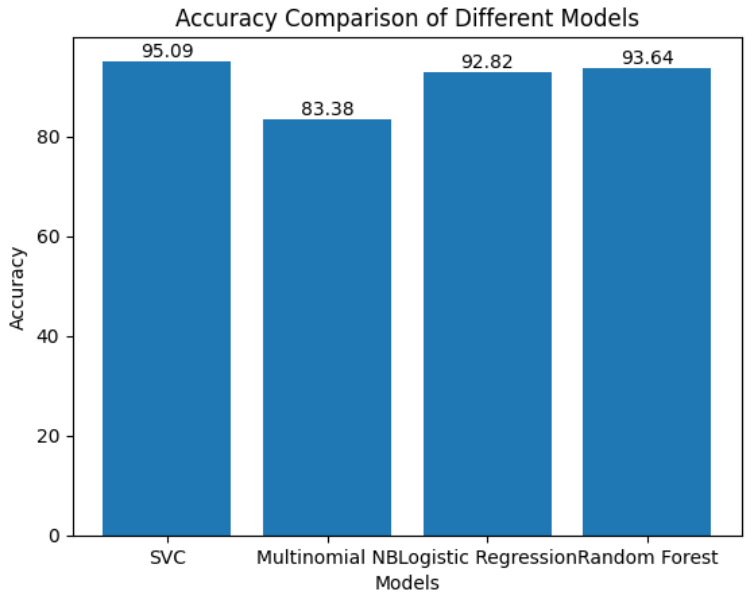

Figure 5.56: English binary classification plot for all models

Bangla (Binary Class) : From the following bar plot we can see that, the Random Forest Classifier is providing the highest accuracy which is about 81%. SVM(RBF) is providing about 80% of accuracy which is higher than both Multinomial NB and Logistic Regression but lower than Random Forest Classifier. Multinomial Naive Bayes is performing lowest among all. It provides an accuracy of about 75%. The logistic Regression Model is performing lower than both SVM and Random Forest but provides slightly higher accuracy than Multinomial Naive Bayes. It's about 76%. Figure 5.57 shows the binary classification plot for all models for the Bangla dataset,

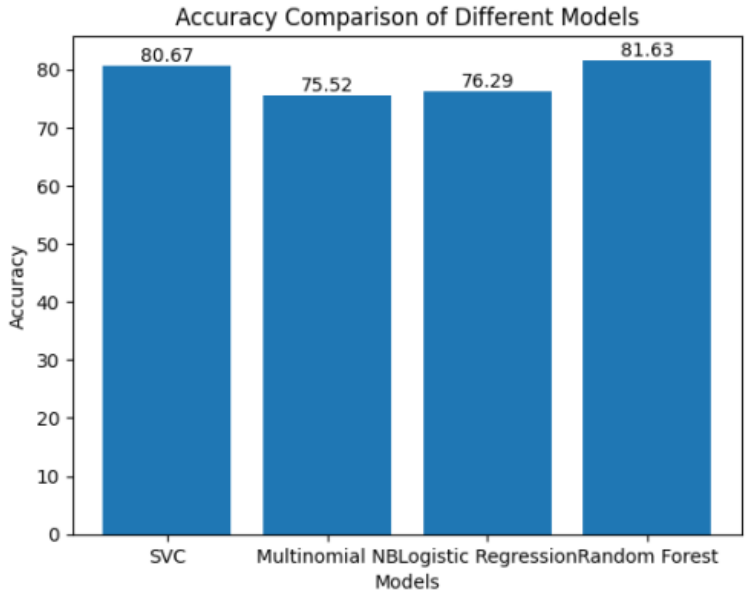

Figure 5.57: Bangla binary classification plot for all models

Bangla (Multi Class) : From the bar plot 5.58, we can see that, the Random Forest Classifier is providing the highest accuracy which is about 68%. SVM(RBF) provides about 65% accuracy which is higher than both Multinomial NB and Logistic Regression but lower than Random Forest Classifier. Multinomial Naive Bayes is performing lowest among all. It provides an accuracy of about 52%. The logistic Regression Model is performing lower than both SVM and Random Forest but provides slightly higher accuracy than Multinomial Naive Bayes. It's about 54%. Figure 5.58 shows the multi-class classification plot for all models for the Bangla dataset.

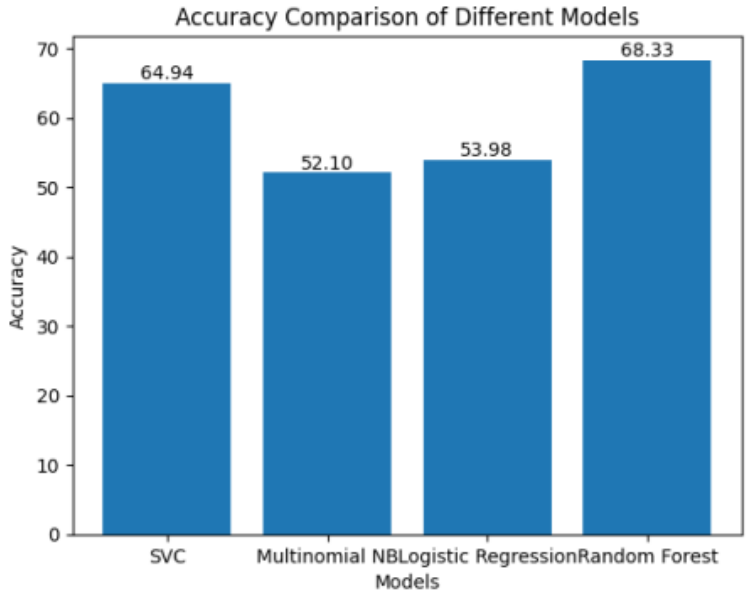

Figure 5.58: Bangla multi-class classification plot for all models

5.3.2 Deep Learning Models

The performance of these seven models which work with multi-class classification is shown in the table. Equally well-balanced and uneven datasets are used in these models. Below is a description of the model's performance analysis:

Table 5.39: Comparison of Models for Multi-class

Model	Balanced Data				Unbalanced Data			
	Acc	Prec	Rec	F1	Acc	Prec	Rec	F1
DistilBERT	0.9087	0.9085	0.9087	0.9079	0.8345	0.8358	0.8345	0.8348
XLM-RoBERTa Base	0.9196	0.9204	0.9196	0.9196	0.8319	0.8343	0.8319	0.8325
BanglaBERT	0.9123	0.9116	0.9123	0.9116	0.8187	0.8184	0.8187	0.8184
LSTM	0.8733	0.8740	0.8733	0.8735	0.7158	0.7182	0.7158	0.7139
Bi-LSTM	0.8669	0.8667	0.8669	0.8663	0.7557	0.7555	0.7557	0.7521
CNN+LSTM	0.8640	0.8704	0.8640	0.8651	0.7695	0.7696	0.7695	0.7668
CNN+Bi-LSTM	0.8663	0.8668	0.8663	0.8657	0.7833	0.7877	0.7833	0.7834

Based on the table above, we can conclude that DistilBERT and XLM-RoBERTa are the two most well-performing models. The f1-score, recall, accuracy, and precision are all good for both balanced and unbalanced data. For balanced data, the accuracy of XLM-RoBERTa and DistilBERT is 0.9196 and 0.9087 respectively. The values for unbalanced data are 8319 and 0.8345. These two models are operating pretty similarly overall. BanglaBERT comes after XLM-RoBERTa Base and DistilBERT. BanglaBERT's accuracy is marginally lower than DistilBERT's for unbalanced data, despite having a higher accuracy overall. For this reason, this model comes in second place to DistilBERT and XLM-RoBERTa. For the balanced dataset, the performance of LSTM and Bi-LSTM is likewise fairly comparable. For LSTM, it is 0.8733, and 0.8669 for Bi-LSTM. However, Bi-LSTM performs better than LSTM for the unbalanced dataset in terms of accuracy and other metrics. CNN+Bi-LSTM and CNN+LSTM come next. For balanced datasets, they perform effectively as well, but for unbalanced datasets, CNN+Bi-LSTM and LSTM outperform them.

BERT-based models (DistilBERT, XLM-RoBERTa Base, BanglaBERT) generally outperform traditional models (LSTM, Bi-LSTM, CNN+LSTM, CNN+Bi-LSTM). BERT-based models outperform other models on a variety of metrics and data types. Although LSTM and Bi-LSTM models function rather well, models based on BERT perform better. CNN-based models (CNN+LSTM, CNN+Bi-LSTM) perform substantially worse than BERT-based models. Consequently, we can say that XLM-RoBERTa is the best of all.

The performance of the three models which work with binary classification is shown in the table below. These models are applied to the newly combined dataset. Equally well-balanced and uneven datasets are used in these models. Below is a description of the model's performance analysis

Table 5.40: Comparison of Models for Binary-class

Model	**Balanced Data**				**Unbalanced Data**			
	Acc	**Prec**	**Rec**	**F1**	**Acc**	**Prec**	**Rec**	**F1**
M-bert	0.92	0.92	0.92	0.92	0.90	0.90	0.90	0.90
IndicBERT	0.93	0.93	0.93	0.93	0.89	0.89	0.89	0.89
XLM-RoBERTa-large	0.84	0.84	0.84	0.84	0.84	0.84	0.84	0.84

Across all three models, M-BERT and IndicBERT balancing the dataset generally leads to an improvement in accuracy, precision, recall, and F1-score. For M-BERT, the improvement is consistent across all metrics. IndicBERT and M-BERT large show improvements in accuracy and F1-score with balanced data. But the XLM RoBERTa has same, precision, recall, and F1-score for both the balanced data and unbalanced data which is not ideal in this case.

Chapter 6

Conclusion and Future Work

In this section, we have described the conclusion and future work of our research.

6.1 Conclusion

For research, we gathered datasets in English and Bangla and categorized them into binary and multiclass groups. Both binary-class and multi-class data have been employed in various models for various purposes. The datasets were prepared using a range of methods to ensure they were ready for analysis. In order to optimize our models, we also looked at feature extraction techniques and applied deep learning and machine learning techniques. We came up with a unique approach to translate English texts into Bangla. By analyzing the performance of models trained on translated English information, we aimed to capitalize on the benefits of cross-language learning. The original Bangla dataset and the translated dataset were integrated, resulting in a large corpus—approximately 55,000 data. Our study encompassed a broad spectrum of architectures, including transformer models, traditional models, hybrid approaches, and a multitude of machine learning and deep learning models. In the case of machine learning, Random Forest (RF) performed exceptionally well in both binary and multi-class classification for the Bangla dataset, while SVM performed well for the English dataset. For the newly translated Bangla dataset, in the case of binary classification, IndicBERT performed the best for both balanced and unbalanced data. On the other hand, for the original Bangla dataset, XLM-ROBERTA performed well in multiclass classification for balanced data. In conclusion, the combination of translating English text into Bangla and exploiting various deep learning architectures is a unique and significant contribution to the field of language-based research.

6.2 Limitations

In our research endeavor, we employed the "banglat5-nmt-en-bn" translator model to facilitate the translation of our English dataset into Bangla, subsequently merging it with our native Bangla dataset. Similar to other deep learning models, the "banglat5-nmt-en-bn" model exhibits inherent limitations in achieving fully accurate translations. Notably, a substantial portion of our English dataset comprises local English language variations, introducing complexities in text translation for the model.

Some cyberbullying texts could not be translated as accurately as possible due to local language and the shortcomings in the translation process. This restriction raises questions about how well our research model will function overall, especially when it comes to accurately representing cyberbullying content in the Bangla context.

It is important to note that we have only run a limited number of models using our newly enhanced dataset as of the research's completion. We have not yet explored the impact of several models on performance inside our experimental framework. It additionally demonstrates how our research is dynamic and always open to new lines of study and development.

6.3 Future Work

In our current research work, we have employed three deep learning models—IndicBERT, xlm-roberta-large, and mBERT—while leveraging an enriched dataset augmented with 10,000 translated English-to-Bangla instances. As we proceed, our study path includes a thorough investigation of several deep learning models. In particular, we intend to use BERT and DistilBERT architectures, Long Short-Term Memory Networks (LSTM), and Convolutional Neural Networks (CNN). Our understanding of model performance and adaptability is intended to be enhanced by this variety. Additionally, we have been employing the enhanced dataset to focus on binary classification. In further stages of our study, we hope to broaden our focus to include multi-class classification. We will manually create new classes in order to accomplish this, enabling a more thorough and detailed examination of cyberbullying incidents. We will tackle the problem of imbalanced datasets in our research for a more reliable and broadly applicable solution. We will use both balanced and unbalanced data in our experiments to make sure that the models perform well across a range of data distributions. Furthermore, we want to investigate binary classification thoroughly for all models that have been used before. This comparison study will provide insights into the respective advantages and disadvantages of each model, creating a more detailed understanding of their suitability for use in cyberbullying detection. We will investigate opportunities for research improvement as part of our continued dedication to developing the field. More

specifically, to expand the scope of our investigation, we will evaluate if it is possible to perform binary classification for each model that is being considered.

In conclusion, in addition to focusing on a wider range of deep learning models and classification scenarios, our future research attempts will also explore the complex issues of dataset balance, offering a more comprehensive viewpoint on the opportunities and challenges in the field of cyberbullying detection.

References

[1] "What is cyberbullying." https://www.stopbullying.gov/cyberbullying/what-is-it. Accessed on April 10, 2023.

[2] "50 alarming cyberbullying statistics to know in 2023." https://techjury.net/blog/cyberbullying-statistics/. Accessed on April 10, 2023.

[3] "Teens and cyberbullying 2022." https://www.pewresearch.org/internet/2022/12/15/teens-and-cyberbullying-2022/#fn-28924-1. Accessed on April 02, 2023.

[4] "Ranked: The 100 most spoken languages around the world." https://www.visualcapitalist.com/100-most-spoken-languages/. Accessed on April 02, 2023.

[5] "Facebook users in bangladesh | april 2023." https://napoleoncat.com/stats/facebook-users-in-bangladesh/2023/04/. Accessed on April 25, 2023.

[6] "Social media trends usages in bangladesh." https://dailyasianage.com/news/46958/social-media-trends-usages-in-bangladesh. Accessed on April 02, 2023.

[7] "Ict4d community newsletter volume 1, issue 4." https://www.biid.org.bd/publications.php. Accessed on April 10, 2023.

[8] "Cyberbullying is an alarming issue during covid-19 situation in bangladesh." https://lawyersclubbangladesh.com/en/2021/08/03/. Accessed on April 10, 2023.

[9] A. K. Das, A. Al Asif, A. Paul, and M. N. Hossain, "Bangla hate speech detection on social media using attention-based recurrent neural network," *Journal of Intelligent Systems*, vol. 30, no. 1, pp. 578–591, 2021.

[10] M. F. Ahmed, Z. Mahmud, Z. T. Biash, A. A. N. Ryen, A. Hossain, and F. B. Ashraf, "Cyberbullying detection using deep neural network from social media comments in bangla language," *arXiv preprint arXiv:2106.04506*, 2021.

[11] S. H. Tuhin, M. Islam, M. Islam, *et al.*, *Cyberbullying Detection using Machine Learning from Social Media Comments in Bangla Language*. PhD thesis, Brac University, 2022.

[12] S. Akhter *et al.*, "Social media bullying detection using machine learning on bangla text," in *2018 10th International Conference on Electrical and Computer Engineering (ICECE)*, pp. 385–388, IEEE, 2018.

[13] M. Jahan, I. Ahamed, M. R. Bishwas, and S. Shatabda, "Abusive comments detection in bangla-english code-mixed and transliterated text," in *2019 2nd International Conference on Innovation in Engineering and Technology (ICIET)*, pp. 1–6, IEEE, 2019.

[14] S. C. Eshan and M. S. Hasan, "An application of machine learning to detect abusive bengali text," in *2017 20th International conference of computer and information technology (ICCIT)*, pp. 1–6, IEEE, 2017.

[15] M. T. Ahmed, M. Rahman, S. Nur, A. Islam, and D. Das, "Deployment of machine learning and deep learning algorithms in detecting cyberbullying in bangla and romanized bangla text: A comparative study," in *2021 International Conference on Advances in Electrical, Computing, Communication and Sustainable Technologies (ICAECT)*, pp. 1–10, IEEE, 2021.

[16] K. Reynolds, A. Kontostathis, and L. Edwards, "Using machine learning to detect cyberbullying," in *2011 10th International Conference on Machine learning and applications and workshops*, vol. 2, pp. 241–244, IEEE, 2011.

[17] M. Dadvar and F. De Jong, "Cyberbullying detection: a step toward a safer internet yard," in *Proceedings of the 21st International Conference on World Wide Web*, pp. 121–126, 2012.

[18] J.-M. Xu, K.-S. Jun, X. Zhu, and A. Bellmore, "Learning from bullying traces in social media," in *Proceedings of the 2012 conference of the North American chapter of the association for computational linguistics: Human language technologies*, pp. 656–666, 2012.

[19] K. Dinakar, R. Reichart, and H. Lieberman, "Modeling the detection of textual cyberbullying," in *Proceedings of the International AAAI Conference on Web and Social Media*, vol. 5, pp. 11–17, 2011.

[20] A. Muneer and S. M. Fati, "A comparative analysis of machine learning techniques for cyberbullying detection on twitter," *Future Internet*, vol. 12, no. 11, p. 187, 2020.

[21] B. A. H. Murshed, J. Abawajy, S. Mallappa, M. A. N. Saif, and H. D. E. Al-Ariki, "Dearnn: A hybrid deep learning approach for cyberbullying detection in twitter social media platform," *IEEE Access*, vol. 10, pp. 25857–25871, 2022.

[22] M. A. Al-Ajlan and M. Ykhlef, "Deep learning algorithm for cyberbullying detection," *International Journal of Advanced Computer Science and Applications*, vol. 9, no. 9, 2018.

[23] M. Raj, S. Singh, K. Solanki, and R. Selvanambi, "An application to detect cyberbullying using machine learning and deep learning techniques," *SN computer science*, vol. 3, no. 5, p. 401, 2022.

[24] N. Romim, M. Ahmed, M. S. Islam, A. S. Sharma, H. Talukder, and M. R. Amin, "Bd-shs: A benchmark dataset for learning to detect online bangla hate speech in different social contexts," *arXiv preprint arXiv:2206.00372*, 2022.

[25] B. Haidar, M. Chamoun, and A. Serhrouchni, "A multilingual system for cyberbullying detection: Arabic content detection using machine learning," *Advances in Science, Technology and Engineering Systems Journal*, vol. 2, no. 6, pp. 275–284, 2017.

[26] C. Raj, A. Agarwal, G. Bharathy, B. Narayan, and M. Prasad, "Cyberbullying detection: Hybrid models based on machine learning and natural language processing techniques," *Electronics*, vol. 10, no. 22, p. 2810, 2021.

[27] "Support vector machine." https://en.wikipedia.org/wiki/Support_vector_machine#cite_note-4. Accessed on May 01, 2023.

[28] "Random forest algorithm." https://www.javatpoint.com/machine-learning-random-forest-algorithm. Accessed on May 01, 2023.

[29] "Convolutional neural network." https://www.geeksforgeeks.org/convolutional-neural-network-cnn-in-machine-learning/. Accessed on November 9, 2023.

[30] "Deep learning. introduction to long short term memory." https://www.geeksforgeeks.org/deep-learning-introduction-to-long-short-term-memory/. Accessed on November 9, 2023.

[31] "What is long short-term memory." https://www.analyticsvidhya.com/blog/2021/03/introduction-to-long-short-term-memory-lstm/. Accessed on November 9, 2023.

[32] "Complete guide to bidirectional lstm." https://analyticsindiamag.com/complete-guide-to-bidirectional-lstm-with-python-codes/. Accessed on November 9, 2023.

[33] "Bidirectional lstm." https://www.codingninjas.com/studio/library/bidirectional-lstm. Accessed on November 9, 2023.

[34] "Introduction to distilbert." https://www.analyticsvidhya.com/blog/2022/11/introduction-to-distilbert-in-student-model/. Accessed on November 9, 2023.

[35] "xlm-roberta-base." https://huggingface.co/xlm-roberta-base. Accessed on November 9, 2023.

[36] "Indic-bert-v1." https://github.com/AI4Bharat/Indic-BERT-v. Accessed on November 9, 2023.

[37] "indic-bert-huggingface." https://huggingface.co/ai4bharat/indic-bert. Accessed on November 9, 2023.

[38] R. R. R.Ramesh Kannan and L. Kumar, "Indicbert based approach for sentiment analysis on code-mixed tamil tweets," *ceur-ws.org/Vol-3159/T3-16.p*, 2021.

[39] "Bert, mbert." https://resource.revealdata.com/en/blog/bert-mbert-and-the-quest-to-understand. Accessed on Nobember 14, 2023.

[40] "Multilingual bert (mbert) model.." https://www.researchgate.net/figure/Multilingual-BERT-mBERT-model_fig2_362263410. Accessed on Nobember 14, 2023.

[41] "xlm-roberta-large." https://huggingface.co/xlm-roberta-large. Accessed on Nobember 14, 2023.

[42] "Cyberbullying classification." https://www.kaggle.com/datasets/andrewmvd/cyberbullying-classification. Accessed on March 20, 2023.

[43] M. F. Ahmed, Z. Mahmud, Z. T. Biash, A. A. N. Ryen, A. Hossain, and F. B. Ashraf, "Bangla text dataset and exploratory analysis for online harassment detection," *arXiv preprint arXiv:2102.02478*, 2021.

[44] "Text lowercase." https://www.lambdatest.com/free-online-tools/text-lowercase. Accessed on May 15, 2023.

[45] "One hot encoding vs. label encoding." https://www.analyticsvidhya.com/blog/2020/03/one-hot-encoding-vs-label-encoding-using-scikit-learn/. Accessed on Nobember 14, 2023.

[46] T. Hasan, A. Bhattacharjee, K. Samin, M. Hasan, M. Basak, M. S. Rahman, and R. Shahriyar, "Not low-resource anymore: Aligner ensembling, batch filtering, and new datasets for Bengali-English machine translation," *10.18653/v1/2020.emnlp-main.207*, pp. 2612–2623, 2020.

[47] "banglat5_nmt_en_bn." https://huggingface.co/csebuetnlp/banglat5_nmt_en_bn. Accessed on November 9, 2023.

[48] "Multinomial naive bayes." https://www.upgrad.com/blog/multinomial-naive-bayes-explained/. Accessed on May 01, 2023.

[49] "Random forest." https://www.ibm.com/topics/random-forest. Accessed on May 01, 2023.

Generated using Undegraduate Thesis LaTeX Template, Version 1.4. Department of Computer Science and Engineering, Ahsanullah University of Science and Technology, Dhaka, Bangladesh.

This thesis was generated on Sunday 26th November, 2023 at 8:28pm.